

# **FINAL REPORT**

(February 24, 2010 – February 23, 2011)

## **“Integrated Reconfigurable Intelligent Systems (IRIS) for Complex Naval Systems”**

**Contract #: N00014-10-1-0629**

### **SUBMITTED TO:**

**Mr. Anthony J. Seman, Office of Naval Research  
(email: Anthony\_Seman@onr.navy.mil)**

**Office of Naval Research  
875 North Randolph Street  
Arlington, VA 22203-1995**

### **SUBMITTED BY:**

**Georgia Institute of Technology  
School of Aerospace Engineering  
Aerospace Systems Design Laboratory (ASDL)  
Atlanta, Georgia 30332-0150**

### **AWARD PERIOD:**

**February 24, 2010 to February 23, 2011**

**February 23, 2011**

### **Principal Investigator:**

**Professor Dimitri N. Mavris  
Director  
Aerospace Systems Design Laboratory  
School of Aerospace Engineering  
Georgia Institute of Technology  
Phone: (404) 894-1557  
dimitri.mavris@ae.gatech.edu**

**20110228255**

# REPORT DOCUMENTATION PAGE

Form Approved  
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Washington Headquarters Service, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington, DC 20503

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS

|   |             |  |                               |  |   |
|---|-------------|--|-------------------------------|--|---|
| 1. REPORT DATE (DD-MM-YYY)<br>23-February-2011  |             | 2. REPORT TYPE<br>Performance/Technical Report |                               | 3. DATES COVERED (From - To)<br>24-Feb-2010 to 23-Feb-2011 |   |
| 4. TITLE AND SUBTITLE<br><br>INTEGRATED RECONFIGURABLE INTELLIGENT SYSTEMS<br>(IRIS) FOR COMPLEX NAVAL SYSTEMS  |             |  |                               | 5a. CONTRACT NUMBER<br>N00014-10-1-0629                    |   |
|   |             |  |                               | 5b. GRANT NUMBER<br>N/A                                    |   |
|   |             |  |                               | 5c. PROGRAM ELEMENT NUMBER<br>N/A                          |   |
|   |             |  |                               | 5d. PROJECT NUMBER   |   |
| 6. Author(s)<br><br>Dr. Dimitri N. Mavris<br><br>Dr. Yongchang Li   |             |  |                               | 5e. TASK NUMBER  |   |
|   |             |  |                               | 5f. WORK UNIT NUMBER                                       |   |
|   |             |  |                               |  |   |
| 7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)<br>Georgia Institute of Technology<br>School of Aerospace Engineering<br>Atlanta, GA 30332-0150  |             |  |                               | 8. PERFORMING ORGANIZATION REPORT<br>NUMBER<br>N/A         |   |
| 9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)<br>Office of Naval Research<br>875 North Randolph Street<br>Arlington, VA 22203-1995  |             |  |                               | 10. SPONSOR/MONITOR'S ACRONYM(S)<br>ONR                    |   |
|   |             |  |                               | 11. SPONSORING/MONITORING<br>AGENCY REPORT NUMBER<br>N/A   |   |
| 12. DISTRIBUTION AVAILABILITY STATEMENT<br>Unlimited Distribution   |             |  |                               |  |   |
| 13. SUPPLEMENTARY NOTES   |             |  |                               |  |   |
| 14. ABSTRACT<br>The report details the progress that has been made by ASDL in developing and applying the IRIS concept for the period of February 24, 2010 to February 23, 2011. The main effort is to develop an integrated dynamic Modeling and Simulation (M&S) environment for understanding the behavior of the next generation of naval ship which is envisioned to be self-assessing, self-predicting, self-planning and self-executing. Progress has been made on three individual tasks associated with resilience analysis and modeling and simulation. A resilience assessment framework was developed to for enhancing the analysis and design of more resilient systems in order to improve the ship survivability and mission effectiveness. The integrated M&S was further improved for evaluating the system's reconfigurability through studying the system's ability for resource allocation and rupture identification and isolation. The graph-based component surrogate modeling approach was implemented to a larger fluid system for damage modeling and survivability analysis. |             |  |                               |  |   |
| 15. SUBJECT TERMS<br>Modeling & Simulation, Reconfigurability, Integrated & Intelligent, Naval Systems  |             |  |                               |  |   |
| 16. SECURITY CLASSIFICATION OF:   |             |  | 17. LIMITATION OF<br>ABSTRACT | 18. NUMBER<br>OF PAGES                                     | 19a. NAME OF RESPONSIBLE PERSON                           |
| a. REPORT   | b. ABSTRACT | c. THIS PAGE                                   |                               |  | Mr. ANTHONY J. SEMAN                                      |
| U   | U           | U  | SAR                           | 65   | 19b. TELEPHONE NUMBER (include area code)<br>703-696-5992 |

# TABLE OF CONTENT

|   |           |
|---|-----------|
| <b>SUMMARY .....</b>  | <b>1</b>  |
| <b>TASK 1: METHODS FOR IMPROVING SYSTEM EFFECTIVENESS THROUGH MORE RESILIENT SYSTEMS DESIGN .....</b>   | <b>1</b>  |
| 1.1. INTRODUCTION.....  | 1         |
| 1.2. BACKGROUND .....   | 4         |
| 1.2.1. <i>Technical Objectives</i> .....  | 9         |
| 1.2.2. <i>Proposed Work for Task 1</i> .....  | 9         |
| 1.3. EXPERIMENTAL RESULTS AND DISCUSSION .....  | 13        |
| 1.3.1. <i>Framework for System Resilience Assessment (Subtask 1.1)</i> .....  | 13        |
| 1.3.2. <i>Framework Demonstration for Resilience Assessment of Naval Systems (Subtask 1.2)</i><br>.....   | 35        |
| 1.4. CONCLUSIONS AND FUTURE WORK.....   | 35        |
| <b>TASK 2: INTEGRATION OF HETEROGENEOUS SYSTEMS .....</b>   | <b>35</b> |
| 2.1 INTRODUCTION.....   | 35        |
| 2.2 GENERAL MODEL SETUP .....   | 36        |
| 2.2.1 <i>Co-Simulation Principles</i> .....   | 36        |
| 2.2.2 <i>Double pendulum: a simple example</i> .....  | 38        |
| 2.2.3 <i>Description of numerical method for time step control of monolithic dynamic systems<br/>                with known equations</i> ..... | 41        |
| 2.2.4 <i>Application of the adaptive time step method to the double pendulum sample<br/>                problem</i> .....                       | 45        |
| 2.3 DISCUSSION OF APPROACH WITH RESPECT TO PROPOSED SUBTASKS.....   | 49        |
| 2.4 RESULTS DISCUSSION AND FUTURE WORK.....   | 49        |
| <b>TASK 3: SURROGATE MODELING OF FLUID COOLING SYSTEMS .....</b>  | <b>50</b> |
| 3.1 INTRODUCTION.....   | 50        |
| 3.2 RESEARCH DETAILS.....   | 51        |
| 3.2.1 <i>Component Surrogate Modeling with Graph-Based Integration</i> .....  | 51        |
| 3.2.2 <i>Damage Simulation of CW-RSAD Surrogate Model</i> .....   | 58        |
| 3.3 FUTURE WORKS .....  | 63        |
| PUBLICATIONS.....   | 63        |
| REFERENCES .....  | 64        |



## Summary

In the design and operation of the next-generation naval surface combatants, more and more attention has been paid to increase the system's affordability, survivability and mission effectiveness. The Aerospace Systems Design Laboratory (ASDL) at the Georgia Tech proposed a framework referred to as Integrated Reconfigurable Intelligent Systems (IRIS) as a solution to satisfy such Navy's requirements. The main effort is to develop an integrated dynamic Modeling and Simulation (M&S) environment for understanding the behavior of the next generation of naval ship which is envisioned to be self-assessing, self-predicting, self-planning and self-executing. The integrated M&S environment is a key enabler for facilitating the system design and operation of the naval complex systems, such as design space exploration, technology evaluation, and scenario analysis.

The following report details the progress that has been made by ASDL in developing and applying the IRIS concept for the period of February 24, 2010 to February 23, 2011. Progress has been made on three individual tasks associated with resilience analysis and modeling and simulation. A resilience assessment framework was developed to for enhancing the analysis and design of more resilient systems in order to improve the ship survivability and mission effectiveness. The integrated M&S was further improved for evaluating the system's reconfigurability through studying the system's ability for resource allocation and rupture identification and isolation. The graph-based component surrogate modeling approach was implemented to a larger fluid system for damage modeling and survivability analysis. New achievements for this period include:

- Defined resilient systems in engineering based on the predefined system resilience characteristics for better performing resilience assessment and design
- Developed a framework of equations for quantitative measurements in support of a resilience assessment method
- Formulated a beam model and implemented the resilience analysis method to it to demonstrate the effectiveness of the framework
- Investigated the methods for time step control for simulation accuracy and computational expense tradeoff
- Proposed an approach for time stepping using an RK23 algorithm and implemented it on an example problem
- Created surrogate models for the YP and CW-RSAD models for damage modeling and simulation
- Conducted analysis based on damage simulation for enhancing survivability of the fluid systems
- Finished 3 journal papers which are under review or in progress

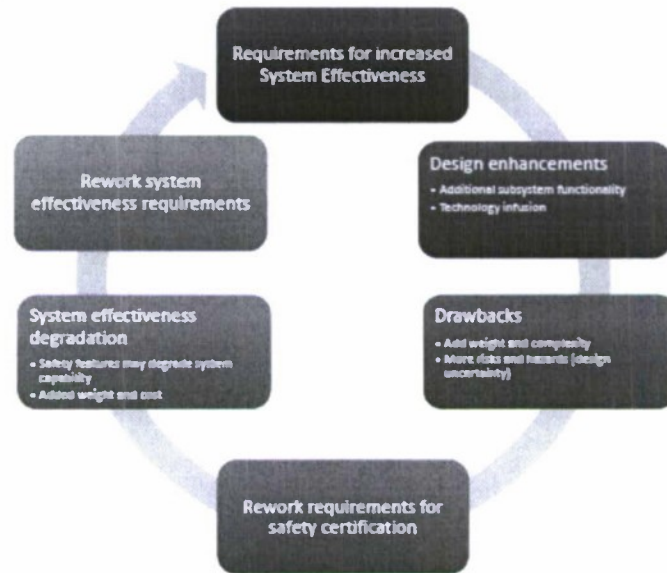
## Task 1: Methods for Improving System Effectiveness through more Resilient Systems Design

### 1.1. Introduction

Current trends in naval ship design have resulted in increased onboard system complexity. Increasing requirements for onboard safety, capability and performance at a



reduced cost have given way to the installation of more advanced engineering systems. Enhancement solutions, such as automation or multi-way communication within subsystems result in a dramatic increase in complexity, which additionally emphasizes the limitations of traditional design approaches.

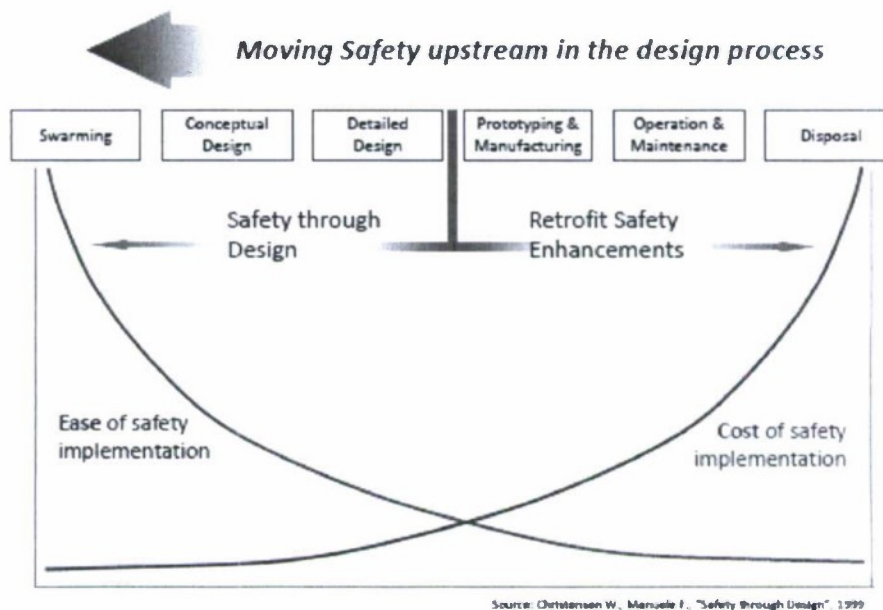


**Figure 1: Complex System Design Cycle: Balancing Effectiveness, Performance And Safety**

Advanced capability translates to increased functionality, with the latter requiring the addition of subsystems for executing the additional functions. Consequently, more subsystems and more internal connections increase overall system complexity. Complexity contributes to operational risk, with the possible emergence of extra modes of failure or support of health or system performance degradation. In conclusion, except for the unavoidable risk around mission operations, increased system complexity itself only increases uncertainty in mission performance, safety and survivability. As Figure 1 explains, increasing the risk on mission performance and system health, leads to the need for revising the design requirements, in order to ensure safety and survivability, while maintaining the desired capability and performance levels.

The problem concentrates on investigating possible methods for improving mission effectiveness, in a way that, the required capability levels are achieved, while safety and survivability are maintained. According to the Defense Appropriation Act of 2004, effectiveness can be improved by including survivability in the design process as a key performance parameter (Rains, 1984). Survivability is a critical safety measure, namely leading to the equal consideration of susceptibility, vulnerability, and recoverability features for ensuring a more robust design. Survivability includes the ability to avoid system detection (Susceptibility) through signature management and to avoid damage (Vulnerability) through survivable design features, e.g. components redundancy or separation.

Safety management is required for exploring all possible options to improve safety and survivability through design. Risk reduction should be substantial under multiple mission environments, also requiring a robust systems design approach as well. Two options are possible, to either retrofit safety solutions at the detailed design phase, or consider safety as a major design objective that is taken into account from the conceptual design phase.



Source: Christensen W., Manuele F., "Safety through Design", 1999

**Figure 2: Moving Safety Early In the Design Process (Christensen, 1999)**

For the first option, safety gaps and risks are identified when the design is almost final, and additional technologies or modifications are incorporated for safety enhancement. There is no guarantee however that the architecture would allow for seamless inclusion of these updates, and it is quite certain that the additional weight will incur extra costs and possible degradation of the performance targets. Hence, a redesign cycle may be necessary, if capability and mission performance requirements are eventually not met. For these reasons, Christensen et al. advocate for the second option, according to which, safety requirements are considered in the early design phases and system survivability is treated as a design objective.

With the IRIS initiative supporting conceptual design methods for more survivable and mission effective naval systems, an opportunity is presented for exploring options on bringing survivability as a design objective. In other words, similarly to how architectures are capable for particular mission performance requirements, one could argue that these architectures are fit for being "inherently" survivable as well, for certain range of mission expectations and environmental risk. Traditional design approaches focus on performance, while safety-driven approaches seek for more robust architectures, usually by technology infusion for safety, automation and intelligence, on a finalized configuration. The great challenge however, is to explore engineering approaches for implementing an equivalent functionality and flexibility on the architecture in parallel to its development for performance, especially given the highly uncertain operational environment that the system is expected to operate.



In the following sections, background information on the current State-of-the-Art in safety and survivability engineering is provided, concentrating on naval applications in particular. These involve relevant safety concepts, assessment methods and design frameworks that are currently used or investigated for further development. The last sections outline the findings related to the research plan for Task 3.

## **1.2. Background**

It has been argued that system effectiveness is a function of system capability, survivability and availability (Habayeb, 1987). Capability is prescribed through the design requirements, and even though mission uncertainty can affect how capable a system is in practice, it is still bounded by theoretical mission expectations. Availability is linked to mission logistics, and is also implicitly linked to system reliability and survivability. Survivability appears to be the attribute that a designer would give priority over other contributions to system effectiveness. It is the attribute that if taken as a design objective, it ensures that the system remains capable and available during the mission.

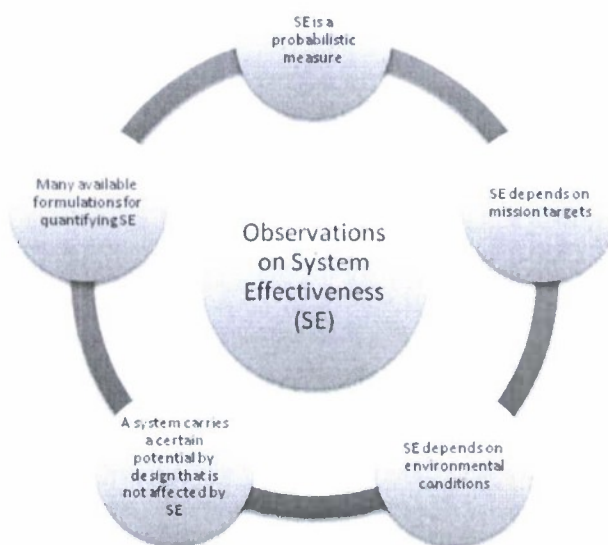
For the military engineering community, there have been several frameworks that recommend design approaches for improving operational survivability. However, not all of them are advocating for early survivability considerations, where the architecture is optimized to accommodate them as well. Given that survivability is indeed enhancing system effectiveness, the fundamental objective for this initiative is to explore alternative and emerging options for improving system survivability, so that system effectiveness is maximized and multi-mission system capability are enabled. For this purpose, a short review of where the State-of-the-art (SoA) is on system effectiveness analysis and survivability-based design is presented.

### *State-of-the-Art in military system effectiveness analysis approaches*

The Weapon System Effectiveness Industry Advisory Committee (WSIEAC) has finalized a methodology for system effectiveness analysis in 1965. The WSIEAC model is based on the enumeration of the significant system states over the entire mission. For instance, a state can be the condition at which a system is available and operable, namely a state in availability and a state in operability can comprise a state at the system level. Concerning metrics that are used to describe these states, there are different approaches in the selected sets of metrics for describing effectiveness, strongly depending on the system point of view. For example, system effectiveness at the mission or the campaign level is typically measured by numbers or percentages of systems killed/survived in battle. At the system level on the other hand, the focus shifts towards individual events and characteristics such as type of system, type of attack, vulnerability, casualties, and recovery time. Hootman (2006) has conducted an extensive literature survey on Measures of Merit (MOMs) for system effectiveness, concluding that the metrics framework by the WSEIAC initiative by the MORS society is very representative of what the SoA is at. System effectiveness is being broken down in sets of contributing metrics that reflect the behavior of the architecture at different and bounded levels.



The Air force Research Laboratory (AFRL) has introduced a probabilistic approach for system effectiveness analysis. It relies on performance-based measures, taking into account the mission objectives and probability of success criteria, as well as system performance under damage propagation that is estimated through companion damage and fail prediction models. The U.S. Navy has also developed a framework for evaluating system effectiveness. A good discussion of this framework is provided by Hanifari et al., through the naval system effectiveness manual NAVMAT P3941-B. As Habayeb also argues, system effectiveness is broken down in availability, dependability and capability. Further breakdown of the three “-ilities” is based on factors that contribute to improving system effectiveness are identified (credits), as well as the ones that add cost to the design of the system (debits). This is a useful first step towards setting up a cost-effectiveness analysis for military operations.



**Figure 3: Observations On System Effectiveness Analysis Approaches**

Figure 3 outlines some of the basic observations on current SoA. A basic observation is that system effectiveness is a probabilistic measure by nature. With the highly uncertain mission operating conditions in the mission environment (Mcmanus, 2007), it is argued that a probabilistic approach is necessary. Deterministic approaches would not capture the variability of external factors (equivalent to noise factors), and that solution robustness could not be guaranteed. This is true for most formulations that are available for quantifying system effectiveness. Indeed, another observation to make is that "system effectiveness" holds different meanings for different communities and applications, while organizations tailor their definitions and methods to apply to very specific problems (Soban, 2001).

To evaluate effectiveness, it is necessary to make an assessment against goals for successful mission completion, but that is not the only requirement. Given that system effectiveness strongly depends on changing environmental conditions under which the

system performs its mission, success is measured by how well the system can achieve its mission goals under varying external or internal conditions. All systems carry a mission success potential and possess a certain level of capability by design. Capability-based design is pre-defining the system performance effect (or system capability) and investigates the solution by generating all possible design alternatives and selecting the optimal (subject to constraints and requirements) that can deliver the same capability. System effectiveness design methods seek for a system that can still be effective at most tasks, even during its post-threat experience segment of its mission. A capable system must also be an effective system, so that built-in capability can be guaranteed, even after experiencing threats, attacks or possible accidents during its mission.

#### *State-of-the-Art in survivability design and analysis approaches*

Survivability-based design has proliferated after WWII, mainly applied on fighter aircraft design. Robert Ball offers great historical insight on how survivability became part of the design for modern fighters and his approach is quite representative of the industry's SoA. His methodology combines survivability assessment methods with a set of susceptibility and vulnerability reduction concepts. The ultimate goal is to examine the impact of possible design enhancements on system survivability and the associated costs of the upgrades. Trade studies determine both the benefits and downgrades in system performance, against the effectiveness and cost for each survivability enhancement feature.

An alternative resource is the Aerospace Systems Survivability Handbook Series, which was developed by the Joint Technical Coordinating Committee for Aircraft Survivability (JTTCG/AS) to provide guidance to government and industry survivability managers, engineers and analysts involved in systems acquisition (JTTCG/ASV1Y2001). The survivability system design and development process follows from a set of stated survivability requirements for a system and evolves throughout the entire system acquisition process. The process begins system requirements and extends through the development, test, and evaluation of an engineering or prototype model of the system.

Switching to naval systems, SoA methods are quite similar to methods used for aerospace vehicles. Most approaches suggest design enhancement strategies, that are based on current susceptibility, vulnerability requirements, but these are methodologies for assessing survivability available, as well. Similarly to (JTTCG/AS), there have been equivalent initiatives within the naval systems community that promote survivability as a design discipline. Regarding naval applications, the OPNAV P-86-4-99 instruction (OPNAV, 1999) is the most popular amongst naval engineers, providing definitions, metrics, and design processes for susceptibility/vulnerability reduction. According to this procedure, survivability is improved by focusing only on vulnerability, implying that susceptibility reduction and recoverability improvement are expected consequences of the former.

Alternatively, discipline-based survivability-based design approaches are focusing on improving total ship survivability by bringing focus on a particular system category. Power systems design is an example, with CPT Doerry's zonal design methods (Doerry,



2007) that emphasize on dynamic survivability by monitoring of the quality and continuity of service (QoS).

Survivability-based design is applicable for the civil sector, with the International Maritime Organization (IMO) and the SAFER-EURORO (EURORO, 2003) initiative for merchant and ferry ships, being the most prominent and commonly adopted. Academia is also supporting the idea of survivability-based design, with researchers introducing their respective recommendations on survivability assessment and enhancement frameworks. Recent efforts include Dr. Sudhoff's survivability-based approach for increasing the operability of naval power systems and the risk and scenario-based survivability design (Brown, 2003) by Dr. Alan Brown.

#### *State-of-the-Art in survivability assessment methods*

All methods discussed previously have clearly demonstrated that is impossible to design for survivability without being able to assess survivability and in some cases test the design, so that a list of necessary enhancements can be identified. In this section, the focus is concentrated on the assessment methods and frameworks available. Most of them have already been mentioned as part of methods for survivability design, such as the kill chain scenario approach, TSSA or IMO assessment. On the more practical side, consulting and engineering companies that work close with military authorities have developed their own environments for survivability based design and assessment, with some (e.g. MOTISS by Alion Technologies) of them to be highly sophisticated stand-alone computer applications that can be used as independent assessment modules for any survivability design methodology.

While reviewing all SoA techniques and methods, some observations have been made, in an effort to understand what research directions one must follow. First, there is no standard design method for survivability design, even though most approaches follow a common logic. Typical survivability enhancement features, such as component redundancy, separation and shielding are immediate techniques that can be properly applied to the design based on conceptual sizing. However, each method is unique in determining the extent of such enhancements and the type of survivability that each enhancement would seek to improve (susceptibility, vulnerability or recoverability).

For multi-mission capable architectures, robust design methods are adopted, which fall under the traditional design approaches are based on optimizing naval systems for performance, based on a limited number of mission scenarios. A robust solution represents a system that in theory would be better prepared to perform multiple mission acts and withstand a larger spectrum of unexpected events. Unexpected however does not imply "never seen before" incident, thus the robust solution is as good as the scenarios or sample incidents, which the architecture is designed to be immune against. In survivability language, robustness is essentially improving vulnerability (probability of kill if hit) reduction. Robustness supports a reactive approach to how the architecture maintains its effectiveness under hazards and environmental mission uncertainty. Robust architectures are usually not characterized by design embedded internal processes, that actively detect and avoid threats or minimize uncertainty, or active reconfigurability



strategies that would introduce internal recovery mechanisms for the case of component kill that may result in total system kill.

### Resilience Engineering

For addressing the shortcomings on current SoA design approaches, new concepts have been investigated, as possible enablers for more survivable system designs. The most promising was that of resilient systems. Resilience engineering is a recent emerging discipline on understanding threats, accident and damage propagation, as well as how systems must be designed to conform to changes that occur around it, for the purpose of withstanding adverse effects and maintaining its mission effectiveness.

For a system to be resilient, it must adjust its functioning prior to or following changes and disturbances so that it can go on working even after a major mishap or in the presence of continuous stress, mainly by being able to be proactive on safety. A resilient system is based on a three-sided functionality. It must be able to anticipate, monitor and respond to environmental change or emerging threats. Resilience can be viewed in conjunction to system safety, survivability, reliability and robustness, and except for a system, resilience can be defined for a structure, a particular material, a network of systems, a communications network or even an organization.

### State-of-the-Art in Resilience assessment methods

Up to present, there have been attempts on quantification of resilience and most of them appear to be system specific and adjusted to requirements and expectations for the design and operation of a particular system. Resilience engineering as a discipline is interpreted by each scientific community, in a fashion that is supporting their own needs on certain system types and applications. Thus, resilience metrics are quite subjective and specific to their application or domain of use.

The lack of a standard analytical or quantitative resilience assessment framework has sparked the interest of individual researchers towards establishing a unified approach (Madni and Jackson, 2009). Except for systems engineering, other possible applications may include to economics and business, or industrial and organizational resilience. Regarding engineering applications, resilience makes sense for materials, structures, infrastructures, architectures and other complex engineering systems. Resilience also applies on networks, such as supply chains or air transportation systems.

A survey of published resilience assessment techniques has been conducted and has revealed possible places to start from for an assessment method. Except for Madni et al. (2009) who propose a generic, yet heuristics based approach on resilience assessment, there have been other domain-specific approaches. Civil engineering appears to be the leading scientific field regarding resilience engineering, with suggested frameworks on infrastructures (Vugrin, 2010), materials (Mitchell, 2009), and indirectly through resilient control for structures (Haddad, 1998). Other applications could include network systems, namely communication networks, or air transportation systems (Reed, 2004).

### 1.2.1. Technical Objectives

As the significance of system safety and survivability has been thoroughly addressed in earlier sections, resilience engineering has been introduced as a conceptual direction for exploring options on enhancing system survivability and thus produces more effective systems. In other words, it should be investigated how all underlying philosophies of resilience engineering can translate into a systems engineering framework.

Yet, system resilience is subjectively defined and several iterations are needed to reach consensus for global definitions, areas of applicability, and assessment frameworks. The latter would be instrumental for integrating system safety and survivability as a product and process characteristic in early conceptual system design. Hence, the main objective with this task is to *investigate the concept of resilience in the context of system safety and suggest a complete framework for assessing resilience in systems engineering.*

With this objective in mind and from the lessons learned through investigating the SoA, the following research questions must be addressed:

- If a resilient system did exist, how would it look like and behave in a dynamic environment?
- How a resilient system does differ from a survivable or safe system?
- What framework for assessing system resilience can be selected and developed in a way to test system robustness and resilience?
- What small scale system exhibits the basic behavioral characteristics of a large scale complex naval system, in order to use it as a canonical problem for the assessment framework development?
- What analysis process should be formulated for associating system resilience to survivability and effectiveness?

As a result of the previous questions, these are the necessary research subtasks that are discussed in the following:

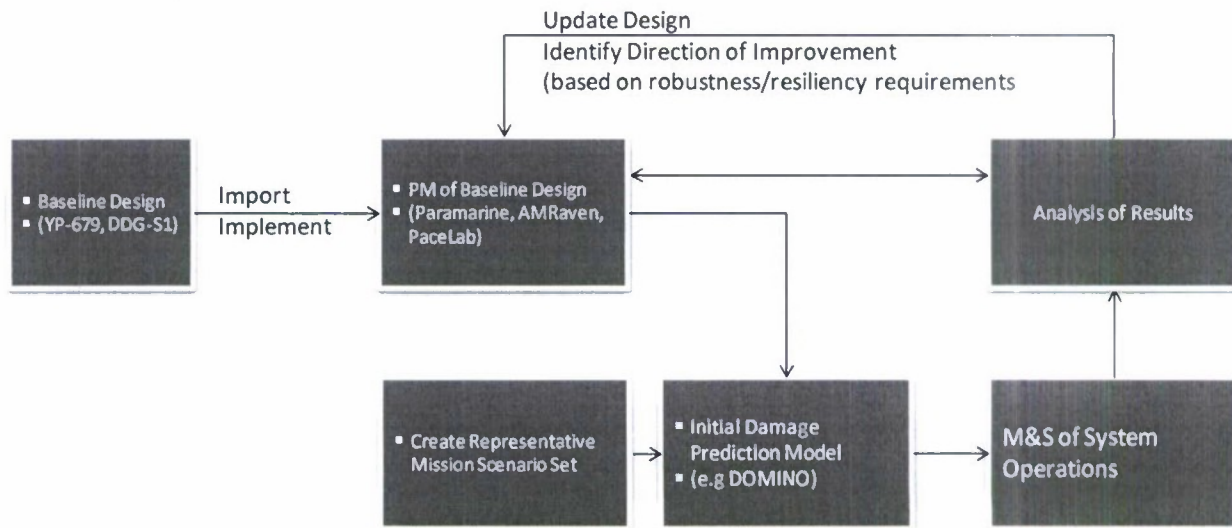
1. Clarification of key words and concepts around system resilience and in contrast to relevant concepts.
2. Proposition of a theoretical framework for resilience assessment, on which a system design methodology will be subsequently built upon. A canonical problem based on a small scale dynamic system is used for development, testing and demonstration purposes.
3. Demonstration of the methodology, using a baseline naval architecture.

### 1.2.2. Proposed Work for Task 1

The ultimate goal of this research initiative is to deliver a systems engineering methodology that allows for the analysis and design of more resilient complex systems. Figure 4 describes a general template of steps towards the acquisition of more resilient designs, based on a naval system application. This generic approach contains the experimentation and design framework, in order to support design space exploration,



systemic damage and accident modeling, and physics-based simulation for system behavior analysis.



**Figure 4: General Template of A Methodology For Resilient Systems Design**

Despite the fact that it could be brought to a point that allows for the design of resilient systems, it only includes “placeholders” for the necessary steps to be taken for resilient design. In other words, if the characteristics of resilient systems are not clarified and the appropriate steps are not included in the method, then it is no different than any systems engineering design method. For instance, adaptability is one of the characteristic features of resilient systems. Reconfiguration is one possible enabler for the degree of adaptability that a system requires to belong under resilient architectures. Thus the method must account for allowing the development and testing of different reconfiguration strategies through intelligent algorithms and selecting the most suitable for a given architecture. Concluding, this leads to a preliminary study, where key words and concepts around system resilience must be clarified and in contrast to relevant concepts.

The originally proposed subtask was suggesting the following:

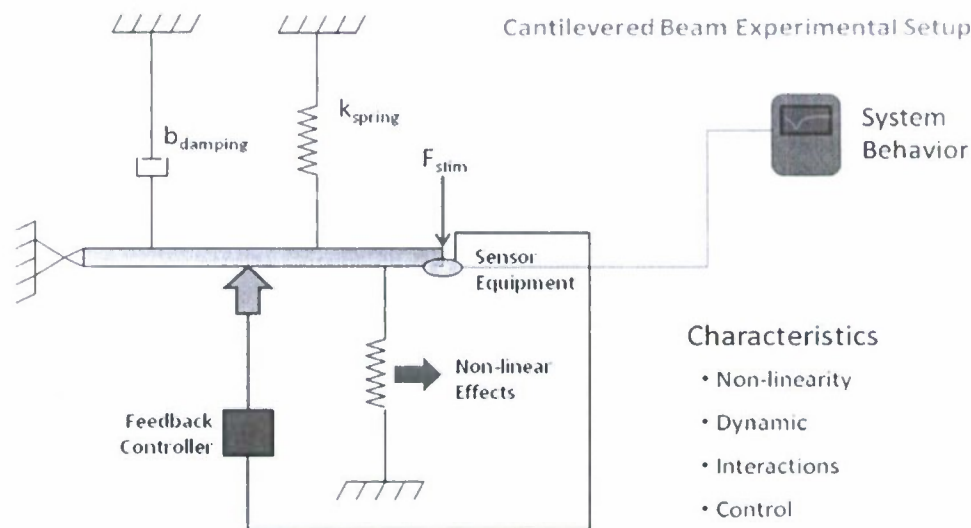
**Subtask 1.1:** Optimize two system architectures, using robust and resilient systems design respectively.

Even though this task would not require an elaborate design methodology for the acquisition of both solutions, it would still require iterations or a verification step that the resilient system actually satisfies some basic resilience criteria. At this time, knowledge about what makes a system resilient or how it behaves is still uncertain, and furthermore, quantitative means of assessing resilience are unavailable. Thus, it has been deemed necessary to revise subtask 3.1 to a new work statement that brings focus on better understanding system resilience and suggests possible ways for evaluating and assessing resilience. The updated work statement is stated as:



**Updated Subtask 1.1:** Define resilient systems in engineering and develop a framework of equations for quantitative measurements in support of a resilience assessment method, based on the predefined system resilience characteristics.

For instructional purposes and for aiding the method development, a small scale “pilot” problem is devised and implemented. The problem involves an elastic/plastic cantilever beam, which is experiencing external dynamic load distributions of varying shape. At the same time, the beam is assumed to withstand a certain amount of uniform load, as its “mission” assignment. As a result of this input stimulus, the beam is experiencing a deflection from its original horizontal equilibrium position, along with a certain amount of curvature that changes its shape. The strain that the beam is experiencing and determines the deflection and the curvature are immediate functions of the beam’s shape, material and the effect of any actuator control system that may be included in the experimental setup. The experimental setup is described by Figure 16.



**Figure 5: Simple Beam Model For Testing The Proposed Resilience Assessment Framework**

Elements that are not part of the beam itself are included to assist the beam in maintaining its equilibrium shape and withstand the destabilizing effects. The material inherently carries similar abilities, however what makes the beam more resilient is the design selection for configuring the external parameters that supports the beam’s mission. As one may understand, the selection of such an experimental setup is anything but random, basically aiming on creating a conceptual and functional replicate of a complex system with similar functions, behavior and characteristics. More about the analogies of the beam pilot problem to a large scale dynamic system are discussed in the following sections.

As part of the theoretical foundation of the resilience assessment approach, a set of responses at different operational levels have been defined and require data that will be

provided by the output of the simulation. Such metrics are the figures of merit for the particular design solution representing the corresponding architecture and will determine its performance based on survivability and mission effectiveness criteria. At the subsystem level, subsystem performance measures can be obtained (e.g. voltage outputs, coolant mass flow rates). Given a scenario per system configuration, system sensitivities and correlations of measures of performance (MoP) to scenario changes can be identified. Such measures are mostly conditional probabilities of achieving an outcome response, given events that occurred earlier, as defined by the scenario event tree analysis. More about the layered structure of resilience measures and the overall effectiveness assessment is discussed in the following sections.

Enhancement of resilience is possible through redundant components, strategic placement of critical components, sophisticated architectural design, lighter materials and enhanced shielding. A resilient architecture must be flexible, and this is possible through a high degree of reconfigurability. Mission reconfigurability is achieved with controllers that support a series of automated functions for sensing, analyzing and selecting an appropriate plan for withstanding and neutralizing the effects of a disturbance. Given that the resilience assessment framework becomes available through subtask 3.1, resilience enhancement concerns are addressed through subtask 3.2:

**Subtask 1.2:** Demonstration of total ship mission effectiveness through resilience assessment, while comparing two architectures under the same mission

With this subtask, the main objective is to demonstrate the resilience assessment framework for naval system architecture. As a baseline, a notional naval ship design is required to be the starting point for the implementation of the method. The common baseline is a version of a Yard Patrol craft (YP). A synthesis and sizing tool is used for generating the geometry and the inner systems distribution. Paramarine is the software that has been used to create this baseline and requires a certain amount of information, such as ship geometry, engineering subsystems, acquisition and operations cost breakdown, mission profiles, threats and hazards and local environmental conditions.

The effectiveness assessment tool has been formulated through a 3-layered combination of Vugrin's resilience assessment tool, the TSSA survivability assessment method and the naval effectiveness evaluation framework (NAVMAT). Initially, the demonstration will only refer to the original baseline. There is an unlimited portfolio of combinations of alternative configurations and mission scenarios that allow for enabling cost-effectiveness tradeoff studies for each solution. For a complete study, the following experiments are necessary:

1. Two configurations (baseline and enhanced) that are assessed for the same mission scenarios.
2. One selected configuration for a series of different mission scenarios

In this demonstration, option 2 is adopted. On a larger scale, both experiments can support the design space exploration and the contraction of the solution space to only a



few selected solutions that will eventually lead to the most resilient design. It should be expected however, that while the resilient solution demonstrates improvement in terms of safety and survivability, it might also incur increased development and maintenance cost.

### **1.3. Experimental Results and Discussion**

Redefining resilience in a systems engineering context is necessary for developing the resilience assessment framework. This framework of metrics and processes is addressed by task 3.1. Work on task 3.2 concludes with the method demonstration results on baseline naval system architecture.

#### **1.3.1. Framework for System Resilience Assessment (Subtask 1.1)**

The main objective of the first task is to further investigate the concept of resilience in the context of system safety and suggest a complete framework for assessing resilience in systems engineering. Given the observed patterns of system behavior, a set of resilience metrics are proposed, essentially capturing the three types of system survivability (susceptibility, vulnerability and recoverability) with respect to the system's dynamic responses. System functionality determines the dynamic responses of highest interest, such as mission performance outputs (e.g. mobility, stability) power delivery (e.g. electric or mechanical power) and system health monitoring (e.g. cooling and temperature levels, structural integrity etc.). Effects from emerging and unexpected adverse behaviors due to changing operating conditions and requirements should also be captured by the framework and included for the resilience assessment.

To break the above objective further down, the following questions need to be addressed and lead us to answers:

- What is a unified definition of the following terms related to resilience: Resilience Engineering, resilient systems and their characteristics?
- How is this definition different than earlier attempts to define system resilience that were more relevant to system safety, reliability, survivability, security, robustness?
- Are there metrics from existing assessment frameworks that could also be appropriate for resilience assessment, based on a given definition and assumptions?
- How could the Goal Question Metric (GQM) method by INCOSE be applied to generate metrics, based on characteristics of resilient systems and their typical expected behavior?
- With a given set of metrics, is there a formal analytical procedure that could be used to demonstrate the goodness of the selected metrics?
- If so, what figures of merit for metric goodness can be employed?
- Is a sensitivity analysis adequate, or more tests would be required to demonstrate the goodness of the metrics?

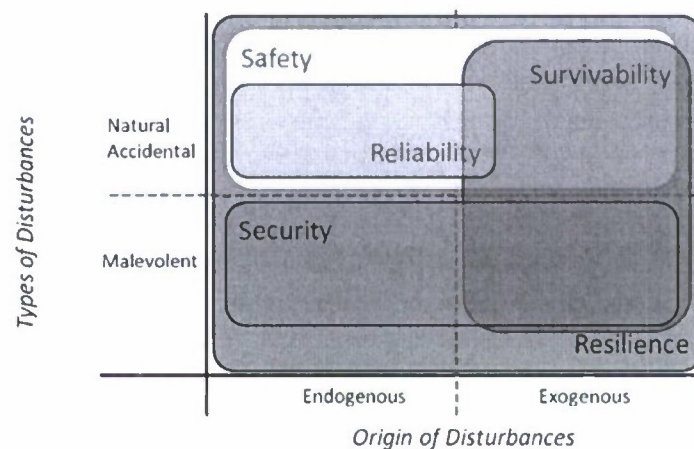
#### *Defining a resilient system in engineering*

Historically, resilience engineering has been regarded as an emerging discipline, initiated in 1973 by C.S. Holling, in a publication regarding the resilience and stability of



ecological systems (Holling, 1973). Madni describes resilience engineering as a discipline that is concerned with monitoring organizational decision making with explicit identification and monitoring of risks (Madni and Jackson, 2009). Amongst other forms, there is economic and business resilience of a financial organization and on a similar realm industrial and organizational resilience. Resilience also applies on networks and on a larger scale, resilience for socio-ecological systems is found.

Holling's definition emphasizes that a resilient system must adopt to change and be able to absorb any adverse effects that result from this change, while maintaining its physical and functional integrity. If these features are mapped against the three types of survivability (susceptibility, vulnerability and recoverability), it appears that adapting and absorbing would fit under vulnerability reduction and maintaining integrity pairs with recoverability. A resilient response by the system would include the ability to efficiently adjust to non-favorable influences rather than to resist them. Such ability could be embedded as collection of internal functionalities and be the basis for certain active features for susceptibility/vulnerability reduction and recoverability increase. It is important to address susceptibility reduction and recoverability for both the system and its mission ability. With this annexation, resilient systems carry extra abilities that would clearly make them more distinguishable from robust systems.



**Figure 6: Safety Concepts And Their Relevance Against Disturbance Type And Origin**

For the purpose of investigating resilience in engineering systems, Holling's definition has been extended to the following:

*System resilience is the ability of the system to monitor sense and warn about an incoming threat, through adaptation to change for effectively absorbing and persisting to adverse effects, while maintaining subsystem connectivity and integrity, preventing system loss and fully recovering so as to perform the system's mission.*

Similarly, *resilient systems* refer to the class of robust systems that carry the ability of actively monitoring, sensing and warning about an incoming threat, through adaptation to change for effectively absorbing and persisting to adverse effects, while maintaining

subsystem connectivity and integrity, preventing system loss and fully recovering the system's mission. Finally, *resilience engineering* is a division of safety engineering that seeks to investigate and propose advanced design concepts and technologies for the analysis and design of more resilient systems. While safety concepts are overlapping when assessed with respect to threat origin and intent, system resilience would be relevant for any kind of threat, as Figure 6 advocates

A resilient system is a robust solution to a great extent, yet it requires the system to be more *proactive* for withstanding and recovering from a threat and its resulting events. To elaborate further on distinguishing resilient from robust systems, a list of associated features has been created and grouped against the three components of survivability. The features were extracted from the earlier definitions of resilient systems. In Figure 7, the degree of the applicability of each feature is qualitatively assessed.

| Survivability type | Resilience definition           | Characteristic   | Robust Systems | Resilient Systems |
|--------------------|---------------------------------|--|----------------|-------------------|
| Susceptibility     | Extended to systems engineering | Ability to monitor threat  | ○              | ⊙                 |
|                    |                                 | Ability to sense threat  | ○              | ⊙                 |
|                    |                                 | Ability to warn about threat   |                | ⊙                 |
|                    |                                 | Ability to adapt to change for more effective system persistence to (adverse) change and system recoverability |                | ⊙                 |
| vulnerability      | Basic resilience definition     | Ability to persist to change   | ⊙              | ⊙                 |
|                    |                                 | Ability to absorb change   | ⊙              | ⊙                 |
| Recoverability     | Basic resilience definition     | Ability to maintain subsystem connectivity and integrity   | ⊙              | ⊙                 |
|                    | Extended to systems engineering | Ability to prevent system loss and fully recover the system's mission  |                | ⊙                 |
| Low ○              |                                 |  | High ⊙         |                   |

**Figure 7: Characteristics Of Resilient Systems And Comparison To Robust Systems**

There is no standard metric for measuring resilience. Moreover, there is no standard analytical method that prescribes experiments and measures for system resilience assessment. Resilience engineering is still at its infancy stage and researchers are currently proposing ideas for metrics and methods. Most approaches are specific to their application or domain of use. For instance, Madni and Jackson (2009) have proposed the following generic types of metrics:

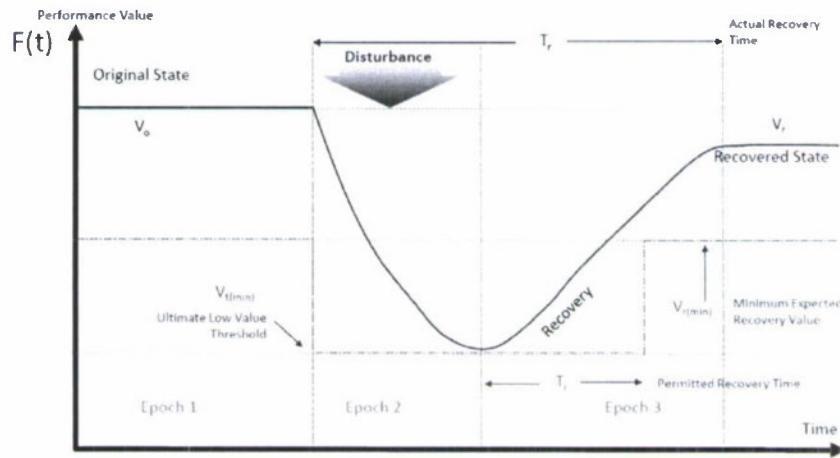
- *Time & cost* to restore operations, configuration, functionality and performance.



- *Degree* to which pre-disruption state has been restored.
- *Number* of potential disruptions avoided.
- *Adaptability* within time and cost constraints.

Hollnagel (Hollnagel et al., 2006) also proposed a similar set of qualitative resilience measures, with time scale being a significant factor, either for disruptions or impact on system recovery. More specifically:

- *Buffering capacity* is a general estimate for system robustness, namely its ability to withstand the change it is experiencing.
- *Self-restructuring capability* is related to the ability to recover from the occurred change.
- *System or subsystem adaptation* as an aggregate estimate of the system's total response to the disruption.



**Figure 8: Typical Dynamic Response To Disruption**

Richards (Richards, 2009) has proposed an epoch-based representation of system response time histories of system response, under disruptions or performance degradations. Having defined the essential functions that represent value delivery and mission accomplishment are time-based, Figure 8 shows a typical system output degradation of system output. The output is the source of characteristic measures that are aggregated to following metrics:

- Time-weighted average performance degradation  $U_L$

$$U_L = U_0 - \bar{U}_T \quad (1)$$

and is the difference between initial utility value  $U_0$  and the time weighted average utility  $\bar{U}_T$  that is

$$\bar{U}_T = \frac{1}{T} \int U(T) dt \quad (2)$$

- Threshold availability  $A_T$

$$A_T = \frac{TAT}{T_{total}} \quad (3)$$

with TAT defined as the average time above threshold.

### System-Mission States

System effectiveness depends on the system's ability to successfully perform its mission and maintain its overall health and integrity under changing external or internal conditions. When a disturbance interferes with normal and stable operating conditions, the system can experience transitions through a set of states. These states are constituted by states that refer to either mission performance or system health. Correlating these states can return a set of system level states.

| Combination | System       | Mission      | Feasibility |
|-------------|--------------|--------------|-------------|
| 1           | OK           | OK           | POSSIBLE    |
| 2           | OK           | FAIL (Rev)   | POSSIBLE    |
| 3           | OK           | FAIL (Irrev) | POSSIBLE    |
| 4           | FAIL (Rev)   | OK           | IMPOSSIBLE  |
| 5           | FAIL (Rev)   | FAIL (Rev)   | POSSIBLE    |
| 6           | FAIL (Rev)   | FAIL (Irrev) | POSSIBLE    |
| 7           | FAIL (Irrev) | OK           | IMPOSSIBLE  |
| 8           | FAIL (Irrev) | FAIL (Rev)   | IMPOSSIBLE  |
| 9           | FAIL (Irrev) | FAIL (Irrev) | POSSIBLE    |

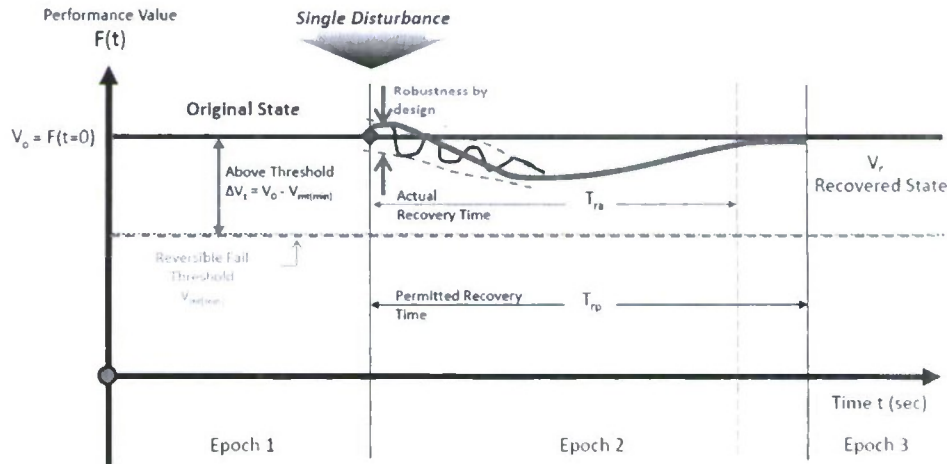
**Figure 9: System Health - Mission States**

To illustrate the previous points, consider both mission status and system health to fall under one of these three states each: *intact (OK)*, *degraded (reversible)* and *degraded (irreversible)*. They refer to the degradation in mission performance ability and system health after the disturbance. The boundaries that mark the transition between states are defined by thresholds associated to system and mission characteristics. Not all combinations of states are feasible though. As an example, Figure 9 displays all possible combinations of states, along with feasibility check. The feasibility check however definitely depends on the system type and its mission requirements.

The states are dynamic, given that they depend on the system dynamic response to a disturbance. In other words, one must analyze the time histories for system outputs, and determine the state according to a set of given thresholds. To illustrate this point, a set of notional response diagrams were constructed. The diagrams could also show the alternative paths that a more resilient system follows, due to its internal abilities of preventing further degradation and recovering. The responses that follow could refer to either the mission ability status or the system's health, so let's assume that they describe the latter.



For the first case, the system maintains its mission performing ability along with its integrity after a single disturbance, as shown in Figure 10. This is the most favorable case where the effects of the disturbance are either weakening until they reach the system or the system is robust enough that it manages to neutralize the effects and become insensitive to the threat. This behavior is an expected feature of a resilient system, yet it is not the sole defining characteristic of it. The degradation is driving the actual performance value below target value; however, it does not go below the *reversible fail threshold*  $V_{mt(min)}$ . This threshold defines a recoverability zone (or safety buffer), within which the system can maintain its integrity and recover by itself.

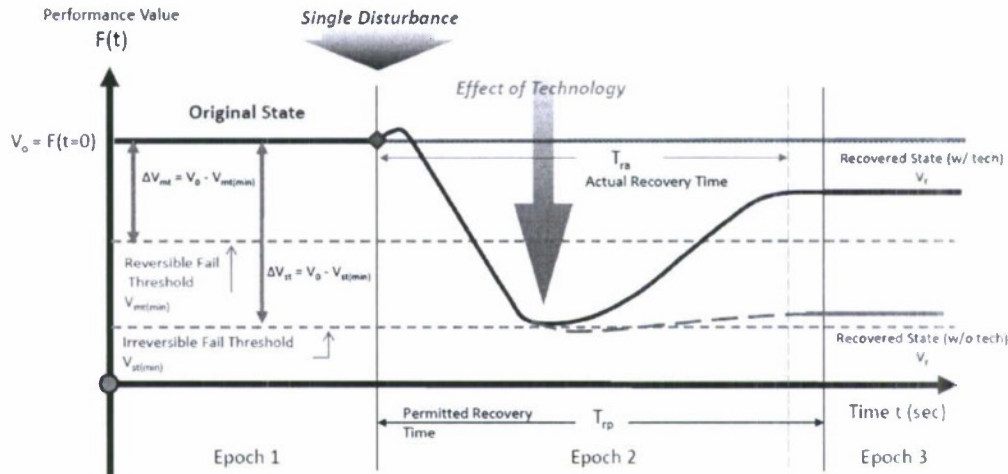


**Figure 10: Reversible Response to A Disturbance**

From the survivability perspective, this degradation diagram describes the vulnerability and the recoverability of the system. In a real example, the system response is not perfectly continuous and smooth. Instead it would be more turbulent or oscillatory, as an outcome of suppressing possible instability through robustness. Overall, this behavior describes a survivable response, with output value restoration to its original level.

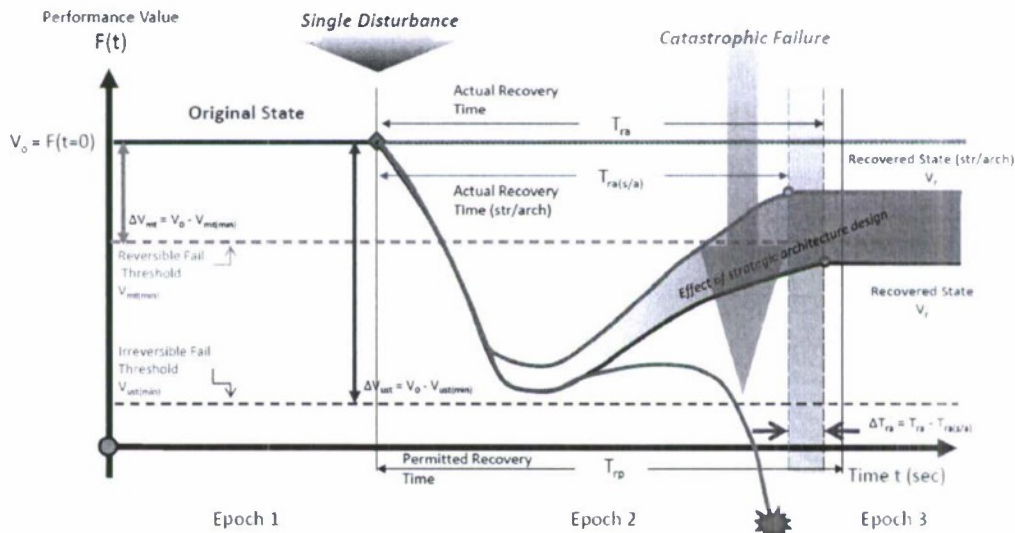
In the second case, the degradation reaches a minimum point below the *irreversibility threshold*  $V_{st(min)}$ . This threshold separates the region where the system can reverse its degradation from a region where it does not have the ability to recover by design. This boundary however is a natural boundary that depends on the certain capability of this design. Figure 11 is showing health degradation that reaches the irreversibility threshold. A conventional architecture would show minimal signs of recovery efforts, but chances are that it would further degrade and experience a catastrophic failure. This outcome is represented by the light blue line in Figure 11.

If the design is more resilient, then it could be capable of overriding the threshold rule and steer itself away from the ultimate system fail threshold. Thus health degradation could become reversible, as it is shown by the dark blue line in Figure 11, in contrast to the expected response of a conventional architecture.



**Figure 11: Partially Reversible Response to Disturbance, With Survivability Enhancements**

The effects of incorporating survivability enhancing technologies into the architecture can prevent the steep degradation. Furthermore, if architectures are designed to be inherently resilient, it may even allow the system to bypass the steep degradations. Survivability enhancements support vulnerability reduction and recoverability, given that the degradation could not be prevented. Resilient architectures that employ reconfigurability and enable adaptability would actively help the system avoid the degradation from the start. The contrast in system health management options, after the system is disturbed are summarized in Figure 12.

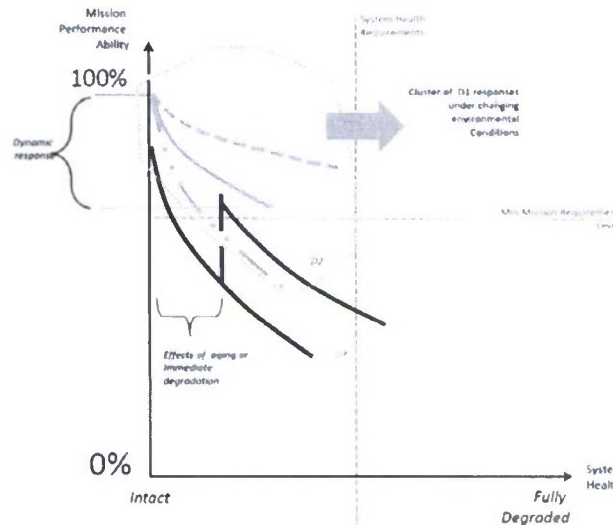


**Figure 12: Irreversible Response to Disturbance vs. Strategic Architecture Design**

Assuming that the dynamic behaviors of mission ability and system health degradation and recovery are similar to those of

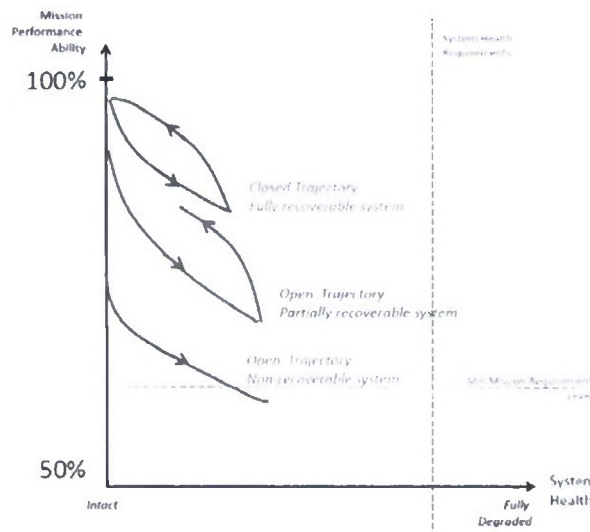


Figure 10 to Figure 12, combined dynamic response "trajectories" can be formed. Similar to scatter plots, where a point in XY space represents a particular design, a trajectory now represents a given system configuration under specified sets of mission expectations and external environmental conditions, as shown in the notional diagram of Figure 13.



**Figure 13: Trajectory Plots for System-Mission Performance**

As part of conceptual design, a Design of Experiments (DOE) can be defined and executed for the data generation and design space exploration. With a particular architecture as a baseline design, the environmental space can also be explored, for the investigating system resilience under combinations of external conditions and mission requirements.



**Figure 14: Trajectories for Various Reconfigurability Levels**

Reconfigurability effects are represented by the type of the trajectory. In a fashion similar to thermodynamic processes, an open loop, one-directional trajectory represents a non-recoverable and irreversible response. A more reconfigurable system can be recoverable,

thus it can result in a bidirectional trajectory. The latter implies partial restoration of original system state, being equivalent to a non-close loop trajectory. Full recovery can be achieved when the steady state value meets the original state value, represented by a closed loop trajectory. More details on reconfigurability and recovery alternatives can be seen in Figure 14.

#### *The proposed set of metrics for resilience assessment*

To support system resilience assessment, there is a need for a systematic procedure that will allow for metric development. The starting point is the defining set of resilience characteristics of Figure 7 and it is required that metrics are associated to them. Resilience characteristics are mapped against the three types of system survivability: susceptibility, vulnerability and recoverability. The development of metrics is based on the GQM approach ("Goal-Question-Metric") that is suggested by INCOSE .

Each characteristic can be assigned to a set of goals that then leads to certain research questions. The breakdown must be clear and straightforward, such that a complete answer can be given through a measure. The corresponding metric of this measure is then a candidate metric that could become part of the framework. Typically, there is a limited number of metric types and these can be classified as:

- *Ratio*: Division of quantities, with the numerator and denominator being mutually exclusive.
- *Proportion*: Division of quantities, with the numerator and denominator not being mutually exclusive and numerator is part of the denominator.
- *Percentage*: Conversion of a proportion in terms of –per hundred- units.
- *Rate*: Represents the dynamic rate of change of the phenomena of interest over time.

In the following, the results of the GQM method are presented, along with discussion on the candidate metrics for supporting resilience assessment.

#### **Susceptibility reduction**

A resilient system must be capable of sensing and warning about incoming threats. Internal mechanisms should capture the presence of a threat, either passively through natural internal processes, or through intelligent detection technologies. Its prognostic capabilities must provide real time indications regarding threat location and direction of propagation for identification of its possible target. These requirements translate to the following functions:

- Ability to monitor and sense threat
- Ability to warn about threat
- Ability to adapt to change

#### *Ability to monitor and sense threat*

The two functions are the basis of system susceptibility. In a threat encounter susceptibility depends on three major factors: the threat, the system, and the mission



(Ball, 2003). According to the mission, the system is required to either monitor emerging threats within its environment, or sense whether the threat is targeting it or has already affected its operations. Traditional systems with sensors measure and estimate the impact of a threat, only after the system is affected and is experiencing change. For instance, a tire pressure monitoring system is using a piezoelectric sensor that can continuously monitor the pressure of a vehicle's tires. If a tire runs flat, it will also notify the passengers that pressure is low on this tire. However, the vehicle does not carry any sensing equipment that can monitor possible hazards on the road (e.g. detect metal objects that can harm the tire) and send warnings.

To address this goal, one may ask what the percentage of threats, hazards and risks that can be monitored in a given range around the system is. The proposed metric would be detectability  $D$  and it is defined as the ratio of number of detected threats over the total number of threats in a given range from the system's position.

$$D = \frac{N_{Detected}}{N_{Total}} \quad (4)$$

Similarly to human detecting abilities, threats can be detected either through their sight in a certain range or by the sound they make, namely the threat's signature. Other detecting options can be the wavelength of a signal that is emitted by the threat, temperature increase due to heat flow rate from the threat or pressure difference due to disruptions to fluid flow rate.

With prognostics having identified the approaching threat, diagnostics must identify affected subsystems and measure the ongoing "change". In particular, real time diagnostics must address how system's mission and health are affected,

Utility functions  $U(t)$  describe system performance output could reflect ongoing system degradations. A generator's power output, power delivery to a load, or heat flow rate on a heat exchanger, all are examples of utility functions that are suitable for diagnostics. For a given utility function  $U(t)$ , degradation rate  $DR$  is the time derivative of performance value loss  $U(t)$

$$DR = -\dot{U}(t) = -\frac{dU}{dt} \quad (5)$$

#### *Ability to warn about threat*

From the point that a threat is monitored up until the moment that it is sensed by the system, all appropriate actions to prepare minimize vulnerability must be taken, as the system experiences change. This warning time period depends on time required for sensor signals to be received by the central monitoring unit, with the slowest signal being critical. A system could potentially be more resilient if this time is minimized and its Responsiveness  $RT$  approaches unity. Normalized with the minimum required time from monitor to sense,  $RT$  is defined as:

$$RT = \frac{\Delta t_{\min}}{t_{\text{sense}} - t_{\text{monitor}}} \quad (6)$$

#### *Ability to adapt to change*

A distinguishing feature of resilient systems is the adaptability to changing conditions. Adaptability is implemented through increased reconfigurability in most cases and results in preventive measures for the anticipated changes. Adaptability could be evaluated with respect to time, namely how much time is needed to fully reconfigure for the change. Moreover, a second impact of adaptability is the shift of performance threshold value to reflect the system's accommodation to change. The threshold is a function of time  $t$  and the impact of the reconfiguration  $a_{\text{rec}}$  namely

$$U'_{\text{th}} = U_{\text{th}}(t, a_{\text{rec}}) \quad (7)$$

The reconfiguration itself is a function of a metric that describes the impact of the experienced change  $\Delta V_{\text{exog}}$ .

$$a_{\text{rec}} = f(t, \Delta V_{\text{exog}}) \quad (8)$$

With clarified states and boundaries, it is necessary to monitor the system's transitions through the states according to the ongoing change. A possible measure of the threshold transitioning is the threshold availability  $A_T$ , introduced by Richards (2009) and is given by equation (3).

#### **Vulnerability reduction**

System vulnerability assumes that the system has already been affected by the threat and all necessary actions to minimize losses and impede damage propagation are enabled. Robust design techniques, as well as vulnerability reduction technologies are the main factors for minimizing the penetration of change to the system's normal operating conditions. In other words, robustness is partially implied through persistence to change or ability to absorb the change in case it is not mitigated. For vulnerability reduction, the system must be able to perform the following functions:

- Ability to persist to change
- Ability to absorb change

#### *Ability to persist to change*

In the context of increased ability to withstand to change, a resilient system adapts to changing conditions in advance compared to conventional systems. The goal is to increase the system's capacity of fighting against the change due to a threat and minimize adverse effects. To support this goal, the rate at which the system's performance is degrading must be identified and degradation rate DR is an applicable metric. The degradation time period  $\Delta t_{\text{deg}}$  is defined

$$\Delta t_{\text{deg}} = t_{U/\min} - t_{\text{deg}0} \quad (9)$$



The degradation time period is the time elapsed from start of disturbance effects  $t_{deg0}$  until the max performance degradation from normal performance conditions  $t_{Umin}$ . For the same time period, the maximum performance degradation  $U_{max}$  is defined

$$\Delta U_{max} = U_{deg0} - U_{min} \quad (10)$$

as the maximum difference between original performance value  $U_0$  and  $U(t)$ .

#### *Ability to absorb change*

Assuming that the change has a non-favorable effect on the system's performance and that the system has not suppressed the threat, it must be capable of becoming insensitive to the unavoidable change imposed by the effects of the threat. Different architectures have different ability of suppressing change. Degradation rate DR is a metric that allows for comparison of architectures. Design improvements on the architecture that would turn it to a more resilient system can be collectively represented by a multiplier  $\rho$  on the original DR, assuming that the improvement effects are linear

$$DR = -\rho \dot{U}(t) = -\rho \frac{dU}{dt} \quad (11)$$

The effects of  $\rho$  can typically represent the effects of reconfigurability. The same effects can reduce the maximum degradation by  $\beta$  namely

$$\Delta U'_{max} = \Delta U_{max} - \beta \quad (12)$$

Given that performance degradation is a dynamic process, a cumulative estimation of the system performance can be provided by the time-weighted average system performance as described by equation (2). The time weighted average performance loss  $U_L$  is given by equation (1). Combining  $U_L$  and  $U_T$ , a measure of the relative loss due to the threat effects can be expressed as a signal-to-noise ratio  $S/N$ , comparing the performance output  $U(t)$  over the time weighted average performance loss  $U_L$  as a ratio. In a logarithmic scale

$$(S/N)_{\log} = -10 \cdot \log_{10}(S/N) = -10 \cdot \log_{10}\left(\frac{U_L}{U_T}\right) \quad (13)$$

#### **Recoverability**

Recoverability of a system includes all necessary procedures the system must employ to recover its health status and restore its mission performance ability. This translates to two responsibilities:

- Ability to maintain subsystem connectivity and integrity
- Ability to prevent system loss and fully recover the system's mission

The first is a robustness feature, but the second is not something that a robust system would be able to do by default. Active reconfigurability and strategic decision making is

necessary for reallocating available resources properly to restore basic and mission related functions.

*Ability to maintain subsystem connectivity and integrity*

Vulnerability reduction can be expressed on a subsystem basis, or otherwise as a collective measure. Effects of reconfigurability are expected to maintain subsystem health status and the necessary connections among them, thus reducing the subsystem loss ratio SLR. This ratio is the number of damaged/inoperable subsystems over total initial number of subsystems:

$$SLR = \frac{N_{damaged}}{N_{total}} \quad (14)$$

System robustness is expected to have favorable effects on maintaining or restoring subsystem connections as part of overall recovery. Similarly, the connection loss ratio CLR is the ratio of the number of damaged/inoperable connections over total initial number of connections

$$CLR = \frac{N_{damaged}}{N_{total}} \quad (15)$$

*Ability to prevent system loss and fully recover the system's mission*

While robustness in some cases can contribute in maintaining system health, it does not necessarily have a favorable effect on mission ability restoration. Resilient systems are expected to restore a system's mission ability close to a target value at an advanced recovery rate. The recovery rate RR at which system health/mission performance ability is restored is

$$RR = \frac{dU}{dt} \quad (16)$$

namely the time derivative of the representative health or mission performance value for the recovery phase. The minimum time required  $t_{rp}$  to reach a restoration steady state is the time elapsed from the performance minimum  $t_{Umin}$  up to  $t_{ss}$  recovery

$$\Delta t_{rp} = t_{ss} - t_{Umin} \quad (17)$$

However, the system might not be fully restored to its original value  $U_0$ . The restoration point offset from the original is:

$$\Delta U_{rec} = U_{dss} - U_0 \quad (18)$$

As the system returns a  $U_{ss}$  value, a final assessment on the system's status requires the comparison to the thresholds at that time point. Remember that thresholds are in general dynamic, thus depending on system reconfigurability and aging.

The total recovery time  $t_{rt}$  is the total degradation time  $t_{deg}$  plus total recovery time  $t_{rp}$



$$\Delta I_{rt} = \Delta I_{deg} - \Delta I_{rp} \quad (19)$$

### System level Assessment

The derived set of metrics is expected to support the complete analysis method that is forming the basis for system resilience assessment. On the next level, the performance responses, along with their target values are combined into overall evaluation criteria (OEC), for constructing collective metrics for resilience.

A state-of practice method to work as template for method development is the Total Ship Survivability Assessment Method (TSSA) . Despite the fact that it refers to ship systems, a generalized version of the TSSA can be augmented to support for resilience assessment for any system. Given the natural association of the concept of resilience to mission uncertainty, this would be a probabilistic implementation.

Metrics are hierarchically distributed, with the *system parameters* as the lower level there are the subsystem metrics. The next level includes the *measures of performance*, or the MOPs that involve conditional probability calculations based upon the values of the SPs. At the higher level, the *measures of effectiveness* (MOEs) are the aggregate metrics for high level mission and system survival assessment.

Calculation of the conditional probabilities for each event outcome is necessary for the MOE aggregate metric estimation that will eventually return the total probabilities of mission  $P_S(\text{Mission})$  and system  $P_S(\text{System})$  survival. According to Ball's classic survivability formulation

$$P_S = 1 - P_K \quad (20)$$

where,  $P_S$  is the probability of survival and  $P_K$  is the killability or probability of not surviving the disturbance. Depending on the type of the disturbance, if this happens to be an attack initiated by an external entity, the equation can become:

$$P_S = 1 - P_H P_{K/H} \quad (21)$$

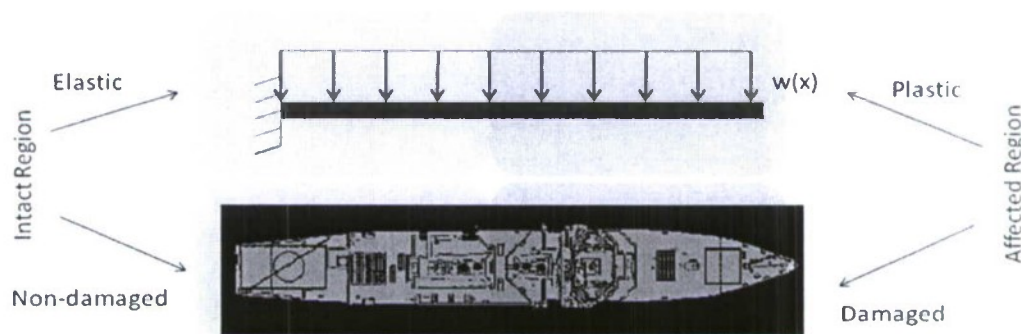
where  $P_H$  is the probability of being detected, also known as susceptibility and  $P_{K/H}$  is the probability of not surviving the attack hit and get killed after being detected, namely system vulnerability. It is interesting to observe that vulnerability is a conditional probability that depends on the outcome of a threat, while susceptibility is a probability that solely depends on whether the threat was encountered or avoided.

The complete mission must generate a scenario that is broken down into epochs and actions. The latter can be accomplished through an *event tree breakdown* (or a "kill chain" according to military engineers), where the entire incident is broken down into subsequent time epochs. The scenario will help define the survivability equation. Susceptibility, vulnerability and recoverability appear as aggregates of the resilience metrics that have been outlined in the previous section. Multiple runs of the same scenario (e.g. in the form of a Monte Carlo simulation) allow for probabilistic estimates

for the survivability components from resilience metrics and then give an overall picture of how the system performs for this scenario.

#### A simple canonical problem

In order to demonstrate the goodness of the proposed metrics, a simple pilot problem has been constructed. It is computational model of a loaded elastic/plastic cantilever. Its mission is to support a uniform load distribution, however an external load distribution applies on the beam unexpectedly, thus increasing its deflection and bring it on its structural limits. Given that the concept of resilience has a wide application for materials, with elasticity and plasticity thresholds to play the same role as the system performance threshold, an analogy has been drawn between the controlled cantilever beam to a complex system architecture that may experience unexpected damage that propagates throughout the system, as argues in

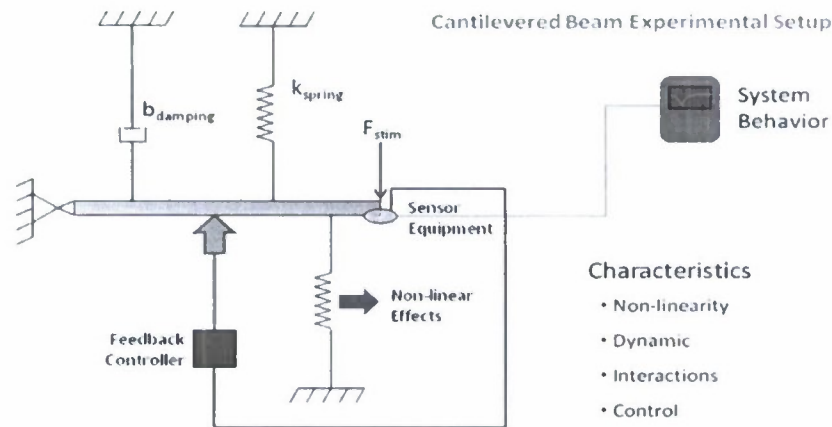


**Figure 15: Philosophical Analogy between the Elastic Plastic Beam and A System Architecture Experiencing Damage Propagation**

The demonstration model involves a cantilevered beam that can experience time varying continuous forces through a weight distribution  $F = w(x,t)$ . As a result of this input stimulus, the beam is experiencing a deflection from its original horizontal equilibrium position, along with a certain amount of curvature that changes its shape. The strain that the beam is experiencing and determines the deflection and the curvature are immediate functions of the beam's shape, material and the effect of any actuator control system that may be included in the experimental setup. The experimental setup is described by Figure 16.

The computational model is a combination of elementary models for either elastic or transient plastic responses. According to the instantaneous value of the bending moment at each beam element, the corresponding moment is "activated" and used for the deflection and deflection rate calculations. The more elements behaving as a rigid plastic body, the greater is the propagation of plasticity. Damage propagation on this system is represented by the equivalent process of the elastic beam turning into a rigid plastic body.





**Figure 16: Simple beam model for testing the proposed resilience assessment framework**

A feedback controller is included to “model” robustness to external unexpected effects. Spring and damper elements included, represent secondary subsystems that the beam as the main center system needs to interact with.

| Type             | State                       | Mission  | Health  |
|------------------|-----------------------------|--|---|
| Elastic Behavior | OK                          | Carry 100% of $w(x)$<br>No disturbance in mission                            | Deflection only due to $w(x)$<br>Bending moment is $M < M_y$ everywhere<br>No plastic hinge   |
| Elastic Behavior | Recoverable degradation     | Carry at least 95% of $w(x)$<br>Disturbance is withstood<br>Mission recovers | Deflection is due to $w$ and external loading<br>$\delta_{max} \leq 0.2L$<br>Bending moment is $M < M_y$ everywhere<br>No plastic hinge       |
| Plastic Behavior | Non-recoverable degradation | Carry at least 95% of $w$<br>Partial or no mission full recovery             | Deflection may exceed $0.2L$<br>Bending moment becomes $M > M_0$ at some beam regions<br>Plastic hinges form until beam becomes rigid plastic |
| Plastic Behavior | Catastrophic failure        | Carry the minimum of $w$<br>Beam breaks                                      | Deflection infinite<br>Rigid plastic beam until it fractures  |

**Figure 17: Beam System States with Mission and System Requirements**

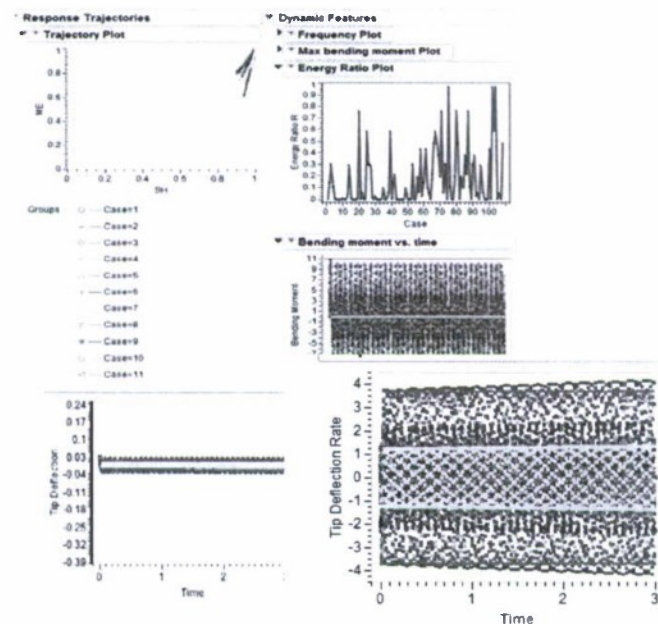
In Figure 17, the detail description of system and mission states is included. Technical parameters for the beam configuration are presented in Table 1.

Technical results at this stage are provided by a JMP analysis environment, based on performance response from the aforementioned canonical problem. At this point, the environment contains the basic necessary functionality, yet its development is not

finalized, as more data visualizations are included. The ultimate goal is to develop an analysis and visualization tool that will be the backbone of the resilience assessment method. Screenshots of the JMP-based tool are shown in Figure 18.

**Table 1: Baseline Configuration Parameters for the Cantilever Beam**

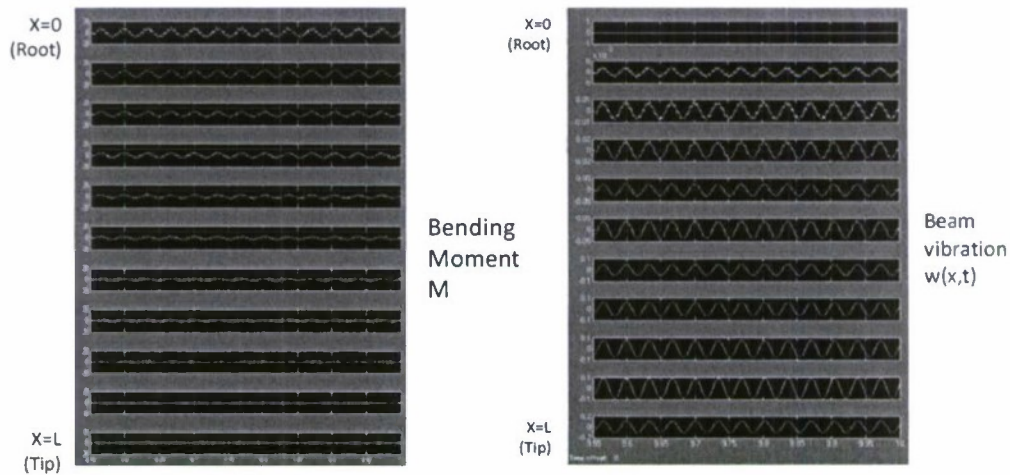
| Beam Parameters            |                        |                        |
|----------------------------|------------------------|------------------------|
| Width                      | $b$                    | 16.3 e-3 m             |
| Depth                      | $H$                    | 4.5e-3 m               |
| Length                     | $L$                    | 0.356 m                |
| Density                    | $\rho$                 | 7850 kg/m <sup>3</sup> |
| Yield Stress               | $\sigma_Y$             | 200e6 N/m <sup>2</sup> |
| Young's Modulus            | $E$                    | 207e9 N/m <sup>2</sup> |
| Max Elastic bending moment | $M_0 = bh^2\sigma_Y/4$ | 16.5 N-m               |
| Control System             |                        |                        |
| Spring constant            | $k$                    | 500                    |
| Damping coefficient        | $b$                    | 25                     |
| Feedback control gain      | $K_p$                  | 100                    |



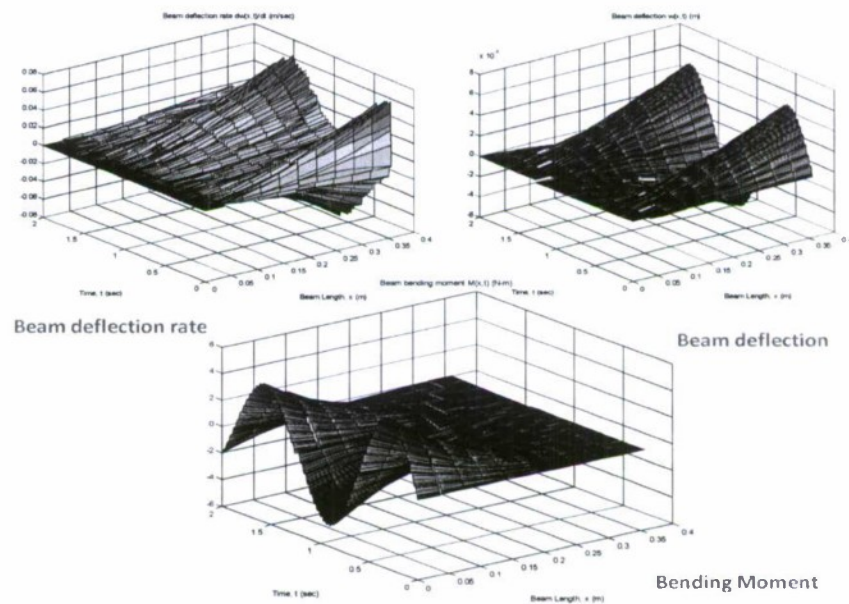
**Figure 18: Screenshots Of The JMP-Based Visualization Tool For System Resilience Assessment**



The computational simulation model was based on the elastic/plastic cantilever beam model by Parkes (1955), including enhancements suggested by Symonds & Fleming (1984). With the first version being inadequate to capture the elastic transition to plasticity, a second version has been developed in Simulink. This version retains the hierarchy of sub-models of the original version, but more effective and accurate theoretical models are implemented for the elastic (Hodges & Pierce) and plastic (Jones) phases. In this enhanced version, the simulation keeps track of vital responses for each beam element. These responses include local deflection, deflection rate and bending moment. Figure 19 presents screenshots of dynamic responses for the beam deflection and bending moment.

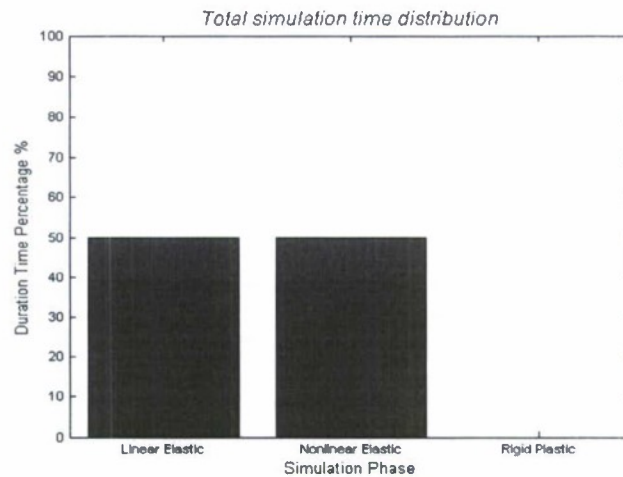


**Figure 19: Screenshots of Beam Deflection and Bending Moment Time Histories for All Elements of the Cantilever Beam**



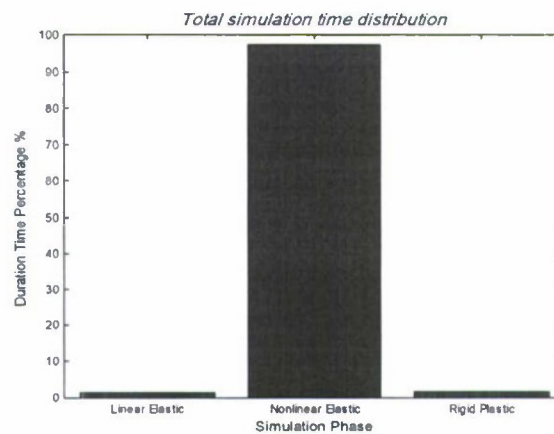
**Figure 20: Beam Simulation Responses for a Disturbance of Small Magnitude**

The disturbance input for this set of experiments is of sinusoidal shape with respect to time and it is linearly distributed over the length of the beam. For small magnitudes of disturbance the beam maintains its oscillatory motion, while it absorbs the energy due to the work that the external load produces. It also successfully maintains its integrity and does not deform plastically. The detailed behavior of the beam over its length  $x$  and over time  $t$  is shown in the 3-D plots of Figure 20.



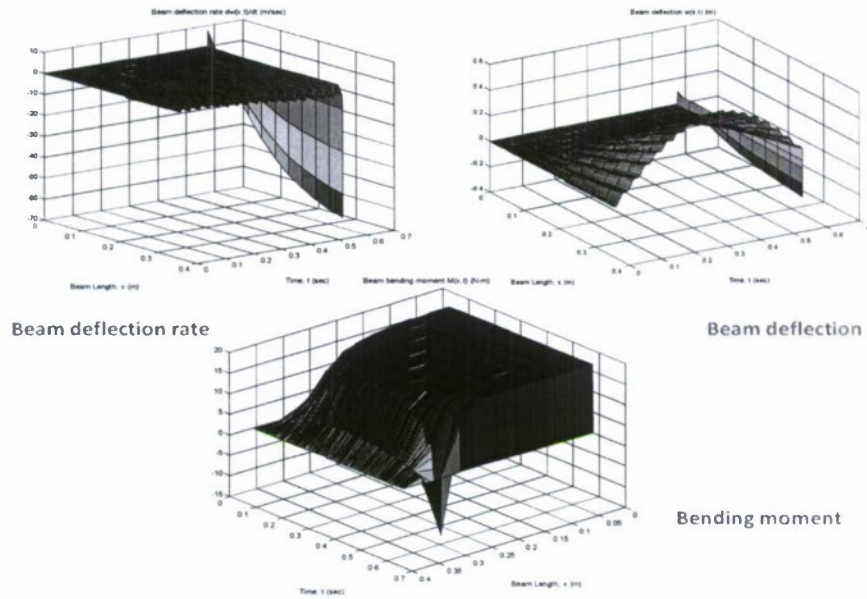
**Figure 21: Time-Based Phase Distribution for Small Disturbances**

Part of the experiment is to keep track of the portion of total simulation time that the system remains on a certain phase. These phases are the elastic, non-linear elastic and plastic behavior phases. To comply with the system analogy, these phases would correspond to system states, similar to these discussed in Figure 17. On the contrary, for larger disturbances, the behavior is different. As one can see in Figure 22, most of the simulation time finds the system in the nonlinear elastic phase and eventually turns fully plastic for a short period of time until it ultimately collapses. System behavior in space and time is shown in Figure 23.



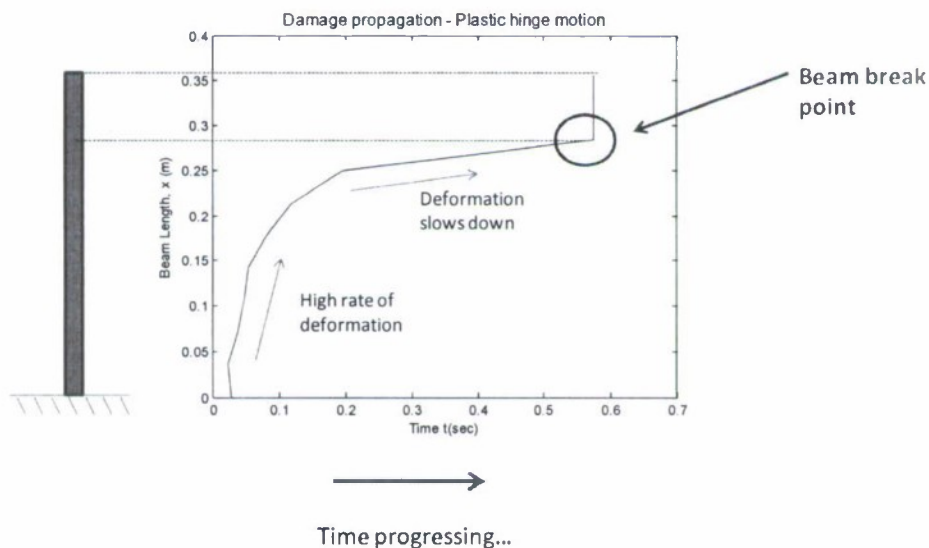
**Figure 22: Time-Based Phase Distribution for Large Disturbances**





**Figure 23: Beam Simulation Responses for a Disturbance Of Large Magnitude**

One of the benefits of this particular implementation is that “propagation” of plasticity over the beam until collapse is thoroughly modeled in the Simulink simulation. That allows for incorporating damage propagation analysis within the framework. As an example, damage propagation can be described in 2-d diagrams of system locations that experience damage over the full simulation time span. The diagram of Figure 24 shows the rate at which plasticity moves from the beam root towards the tip. At the beginning of the simulation, plasticity rate is high but it gradually slows down as more beam elements reach the fully plastic moment threshold. Eventually, plasticity has spread throughout the entire beam and the beam collapses at a certain element.



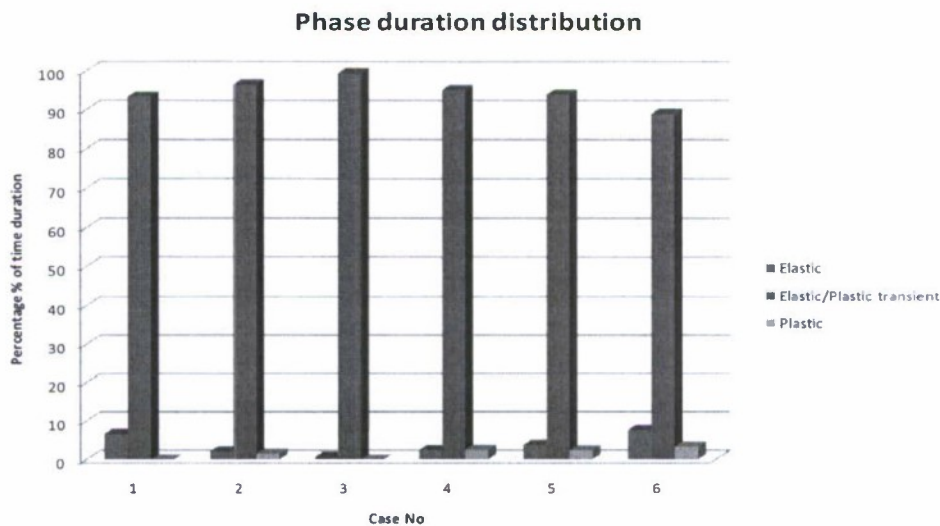
**Figure 24: Plasticity (Damage) “Propagation” Over Time throughout the Beam**

As part of the ongoing efforts to complete the resilience assessment framework, DOE cases are constructed and are used as inputs to the beam simulation for generating data. The data is the pilot for ensuring that the metrics presented earlier are the appropriate ones and sensitivities due to environmental changes and configuration variation are captured. Furthermore, all parts of the process will be compiled to a method with certain steps and experimental outcomes for further analysis. As an example, the cases shown in Table 2 were considered:

**Table 2: Various System Configurations**

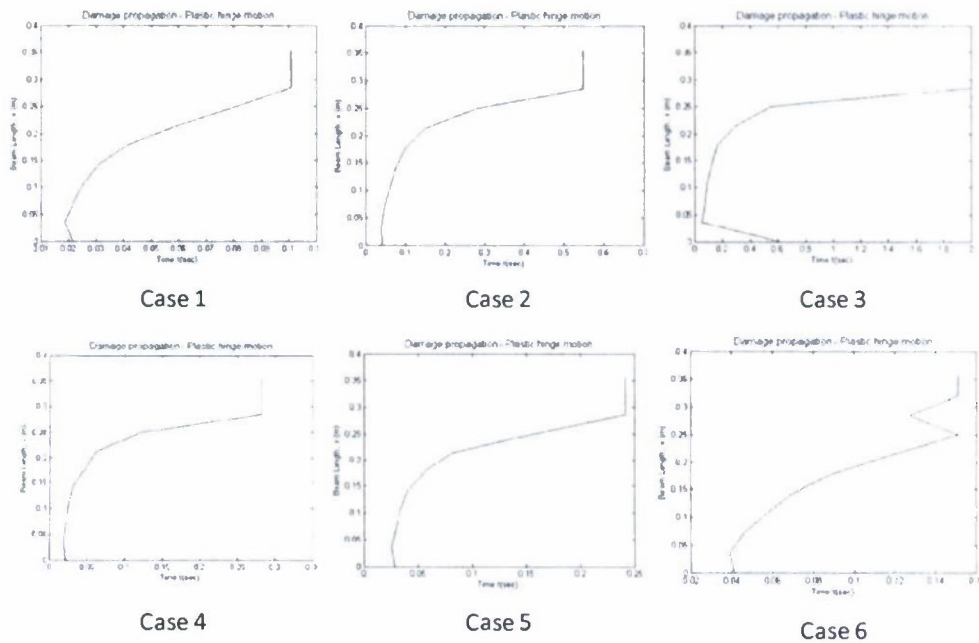
|        | Load<br>distribution<br>magnitude $p_0$<br>(N) | Disturbance<br>amplitude $d_0$<br>(oscillatory) (N) | Disturbance<br>duration $\tau$ (sec) | Spring<br>constant<br>$k_{\text{spring}}$ | Damping coefficient<br>$b_{\text{damp}}$ | Controller gain<br>$K_p$ |
|--------|--|---|--------------------------------------|---|--|--------------------------|
| Case 1 | 4  | 14000   | 0.1                                  | 0   | 45                                       | 40                       |
| Case 2 | 6  | 8000  | 0.2                                  | 0   | 0  | 20                       |
| Case 3 | 6  | 4000  | 0.2                                  | 0   | 0  | 0                        |
| Case 4 | 4  | 12000   | 0.1                                  | 0   | 0  | 40                       |
| Case 5 | 2  | 10000   | 0.1                                  | 0   | 15                                       | 60                       |
| Case 6 | 6  | 10000   | 0.2                                  | 0   | 30                                       | 40                       |

Time duration distributions for each dynamic phase are summarized in Figure 25 and similarly the damage propagation plots were obtained, as shown in Figure 26. In order however, to conduct a full resilience assessment, a larger number of cases-scenarios need to be constructed and simulated. The method must be probabilistic in order to capture the uncertainty around when and how disturbances occur. Part of the ongoing research thus is to explore options on linking the lower level deterministic measures to probabilistic estimates for survivability and furthermore to overall system effectiveness.



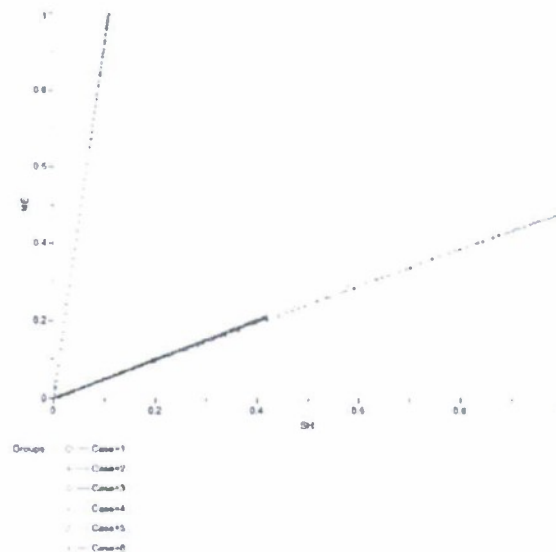
**Figure 25: Time-Based Phase Distribution for 6 Cases**





**Figure 26: Plasticity (Damage) “Propagation” Over Time for Six Experimental Cases**

Given that the resilience assessment tool is not intended only for analysis, but also in support for conceptual design and decision making, a series of visual aids is considered for allowing the down selection of possible design configurations, as resilience and survivability are the design objectives. As an example of such aid, are the mission-health trajectories that describe how the system dynamically evolves towards states that imply either degraded health or performance ability. A link of such instrument to overall resilience assessment is also under investigation.



**Figure 27: Trajectory Plot Based On Six Beam Simulation Cases**

### **1.3.2. Framework Demonstration for Resilience Assessment of Naval Systems (Subtask 1.2)**

For this subtask, the purpose is to use the resilience assessment tool, as this is being developed on the cantilever beam, on naval ship systems architecture. It has been a challenging venture to identify and formulate a suitable modeling and simulation environment for this purpose. Up to this point, a simulation environment for a YP-679 ship model is under development in order to be used as a demonstration enabler for the resilience assessment framework. Regarding the development of the framework, further work is required on processing of the metrics and mapping lower level metrics to probabilistic MOEs.

Regarding the application of the resilience assessment framework in a naval system related problem, the plan is to combine individual models of engineering subsystems to produce integrated models for dynamic simulation. The most significant part of this particular effort will be a routine that models and investigates damage propagation on a naval system. The damage model engine will analyze (damage prediction) and visualize (on the Paramarine ship model) the damage propagation throughout the particular architecture. In this task, a total ship systems operations M&S environment is the desired outcome, including an investigation of damage generation and propagation.

### **1.4. Conclusions and Future Work**

For the remainder of the resilience assessment tool development, the method must be vertically integrated to survivability measures, as well as to provide overall system effectiveness measures. Another possible addition is a cost effectiveness analysis module, through which the designer is allowed to test and evaluate possible architecture enhancements or technology infusion. The latter appears to fit better into the ship demonstration example, given that cost estimates are more accessible and meaningful. In the end, it is expected that while resilient systems should be more mission capable and survivable, this would not come at no cost and thus a cost effectiveness study would offer more insight as to whether the concepts of resilience engineering should become the paradigm for more safe and survivable designs.

## **Task 2: Integration of Heterogeneous Systems**

### **2.1 Introduction**

The new generation of Navy ships will be all-electric, fully integrated, and will be comprised of several main subsystems, such as electrical, hydraulic, and control systems. It would be too expensive and time-consuming to test-build these systems and integrate them as hardware. Also, it would be impossible to test such physical systems under real conditions, especially when they are to be operated under war scenarios. Thus, computer models will be used to simulate the integration and investigate their behaviors. The subsystems are represented by dynamic system computer models that accurately represent the actual physical systems. An integration environment links the sub-system models into a single dynamic model that accurately represents the real world. The system co-



simulation is performed using the time-discrete method to make sure the right models get the right data at right time. In this study, the sub-models are executed in small and equal system time steps. After their execution, they exchange their output values among each other. The selection of this system time step is of critical importance, since it determines the accuracy of the simulation results and the computational expense that needs to be provided. This report presents the research that has been undertaken with respect to the time stepping of co-simulation where the equations for the integrated sub-systems are unknown. A discussion of the three previously proposed subtasks is included.

## **2.2 General model setup**

The next generation Navy ship follows a new design paradigm, shifting towards an all-electric and fully integrated system. A key enabler for this paradigm shift is the accurate and timely modeling and co-simulation of the implemented sub-systems. For this, an integrated modeling and simulation framework is developed, which will integrate different models of physical dynamic systems into an overall model that accurately represents the integrated real system.

The new ship systems to be integrated into the model are more complex and complicated than previous generations of systems, and their development is becoming more costly and time-consuming. Therefore, modeling and simulation (M&S) becomes an imperative part of the system development process, since it enables the designer to integrate the models of the developed system, which in turn allows system investigation and error determination before an actual physical system is produced. This results in greatly reduced time and cost efforts necessary during the design process. Therefore, it becomes increasingly important to develop accurate modeling and simulation tools that help to enable sophisticated and accurate M&S tasks.

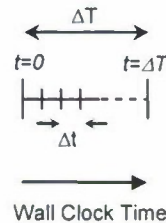
Even though modern computers provide unsurpassed calculating power, simulation models become more and more complex at the same time. Furthermore, the modeling paradigm shifts towards a distributed simulation of integrated sub-models. Some reasons for this are use of legacy codes, distributed computing, and interdisciplinary modeling. All these lead to co-simulation of dynamic systems as a means of interconnecting different dynamic sub-models into an overall integrated model. It is desirable to find ways in which the computational expense for such complex simulations can be reduced without sacrificing accuracy. Ideally, this approach will also provide some sort of error control, so as to be able to see how the simulation results vary as options for saving computational expenses are applied.

### **2.2.1 Co-Simulation Principles**

The common approach of integrating the underlying continuous dynamic systems is by implementing a discrete time step simulation. Commonly used computers are digital, and hence cannot process data in infinitely small steps. Therefore, any simulation will have to be discretized, either in time or in state. The most common approach to simulating continuous systems on digital computers is to execute the digital model at discrete times, employing finite time steps. For one single model, it generally consists of a "global" system time step which determines the overall system time frame in which the model



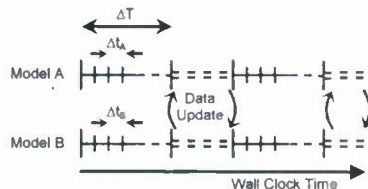
behavior is to be simulated. This time step is henceforth denoted as  $\Delta T$ . In order for the simulation to obtain the results after the global time step  $\Delta T$ , the simulation will start to process the model with the given initial conditions. However, within the time frame from the initial to the final time, the simulation can employ one or several sub-steps to reach the end of the simulation. This “local” time step  $\Delta t$  can be as large as the global step  $\Delta T$  (this represents an upper limit), or smaller, thus splitting up the execution step into smaller sub steps. Figure 28 depicts the time step scheme.



**Figure 28: Model Time Stepping Schematic**

One challenge for a continuous system simulation is to choose an optimal trade-off between local and global time step size. If the global time step covers a long system time, a simulation in one time step will result in very inaccurate results. Hence, the local time step  $\Delta t$  will have to be chosen smaller than the global step  $\Delta T$ . The smaller  $\Delta t$  gets, the more accurate the simulation becomes in theory. However, this comes with some disadvantages. Firstly, reducing the local step size will increase the number of times the simulation has to calculate the model, thus increasing the required computational expense and real-world time requirement. Secondly, the smaller the simulation time step becomes, the larger the calculation errors become due to the limited accuracy that a digital computer has to represent numbers. The calculation errors will sum up over the run of the simulation, and eventually the result will become too inaccurate to use, or the simulation itself will cease to work due to these errors. It is therefore clear that the time step used must be traded off between fastest execution and highest accuracy. As a result, an optimal time step setting must be found. This setting will depend on the underlying model's dynamic properties.

When co-simulating several dynamic sub-models to represent an overall dynamic system, the different dynamic behaviors of the sub-systems need to be accounted for. It must be made sure that all systems start at the same simulation time, and with initial conditions referring to that time, and end after the same time step. Only then can the sub-systems exchange their output states with the other systems. Figure 29 shows the execution schedule for such a model.



**Figure 29: Distributed Model Execution Schedule**

In this setup, it is assumed that the sub-models are distributed over different computers, and are executed in parallel. Once all sub-models have reached the end of one calculation step  $\Delta T$ , execution is halted, and results are exchanged between the models.

The timing of the sub-model execution, and the data exchange between them, is the task of a scheduler, which is the main component of a co-simulation environment. The scheduler is responsible to make sure that the simulation runs the sub-models from a common starting time to a common end time. The scheduler also exchanges the data between the models after each time step, and makes sure that each model gets only those outputs as new inputs which it actually requires. The scheduler also enforces any constraints between the sub-models, saves the required responses into a database, and ensures data consistency. The scheduler is usually written using programming languages (e.g. C/C++, Java) or scripting languages (e.g. VBScript in ModelCenter®). This offers more flexibility in the design of the scheduler, and speed in the execution of the simulation, but also requires a thorough program design.

The scheduler saves all sub-system responses after every time step. Data is taken from here for every data update the scheduler issues. The saved data can be used to visualize and monitor the different states and responses. Using a database for this task reduces computational expense since the data does not have to be recreated through co-simulation execution.

The sub models used in a co-simulation are treated as “Black Boxes”. They are assumed to be provided by third party contributors. Hence, their internals are confidential and cannot be changed. The only information these models require are its state inputs at the beginning of each simulation time step, and the only information obtainable from these models is the state information at the end of each simulation time step. It follows that the co-simulation environment needs only to handle these variables.

The local time step, referred to as  $\Delta t$  in Figure 29, is the time step used by the sub model’s solver for the numerical integration. The local time step does not necessarily have to be fixed for all sub-models, neither does it have to be fixed within each sub-model.  $\Delta t$  affects the solution stability of the individual sub models during co-simulation.

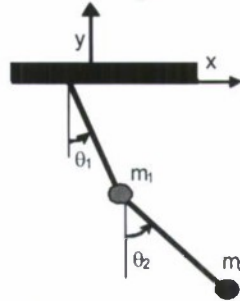
The sub-models run for a system time  $\Delta T$ , after which they stop execution and exchange their variables. This time step must be greater or equal to  $\Delta t$ .  $\Delta T$  affects the stability of the sub-model, but also the accuracy of the results of the co-simulation. Basic information on modeling and simulation can be found in [Zeigler, Kim and Praehofer 2000].

### **2.2.2 Double pendulum: a simple example**

In order to be able to evaluate co-simulation principles, specifically an adaptive time stepping algorithm using a simple model of a 2D double pendulum was created. This model is well-understood and widely referenced. A double pendulum is a system of two pendulums that are tightly coupled. One pendulum bar is attached to a fixed point, and the second bar is attached to the lower end of the first bar at the first bar’s mass point.

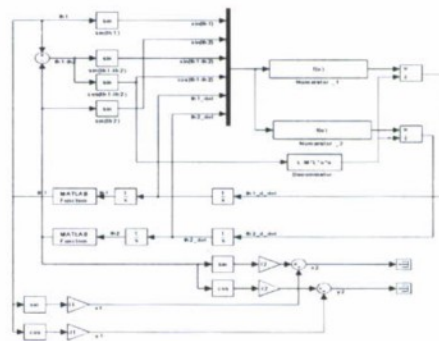


The overall system's dynamic behavior depends on the two masses, the lengths of the two bars, and the initial conditions. The bars are assumed mass-less, and no friction is present. Figure 30 shows a schematic double pendulum.



**Figure 30: Double pendulum schematic**

The system equations describing a double pendulum are a set of two second order nonlinear differential equations, each corresponding to one of the masses. A monolithic computer model of the whole system (representing the two nonlinear equations) was created using Matlab/Simulink®. A block diagram of the Simulink® model is shown in Figure 31. For later examinations, this model is used as the reference to determine the deviations in the results of a co-simulated model.



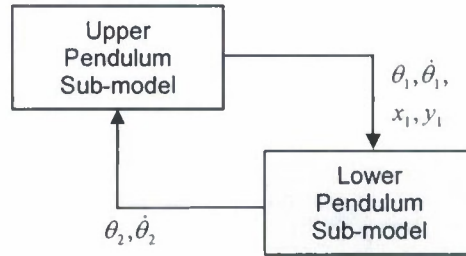
**Figure 31: Simulink Model of Double pendulum**

The monolithic model was split into two sub-models, each of which represents one of the pendulum bars. Hence, it could be used as the basis to which the co-simulation was compared. Initial runs were made to determine the overall system behavior, and analysis can be done about how the time step  $\Delta T$  influences the results.

Co-simulation adds another level of complexity to the overall time step problem. In order to create a simple model with which the time step variation algorithm could be developed and tested, the monolithic model of the double pendulum in Figure 31 was split up into its two subsystems, each representing one of the pendulum bars. These two sub-models were then integrated into a simple co-simulation model using a Matlab® script. The script is responsible for calling the sub-models, synchronizing the data exchange between them, and adjusting the time step. Using this script, trade studies were performed to evaluate variable time step algorithms and their effect on simulation execution speed and



accuracy. The investigations were made on the lower pendulum subsystem. Figure 32 shows a schematic of the split model.



**Figure 32: Split Pendulum Execution Schematic**

In order to co-simulate the split model, it must be ensured that all models start and end at the same global simulation times. Hence, for all models,  $\Delta T$  must be the same, and the co-simulation must start at the same system time for all sub-models. It must though, also be ensured that all models do the same number of sub-steps  $\Delta t$  in this period. For the scope of this research, global and local time steps are kept equal at all times. The task then becomes to find a time step  $\Delta T$  that provides the best compromise between execution time and deviation error.

Initially, a time step can be chosen according to several aspects, such as experience, results from single component simulations, system frequencies, etc. A minimum step size should be defined, which represents a lower bound due to system natural frequencies, and will not be exceeded by the variable time step algorithm.

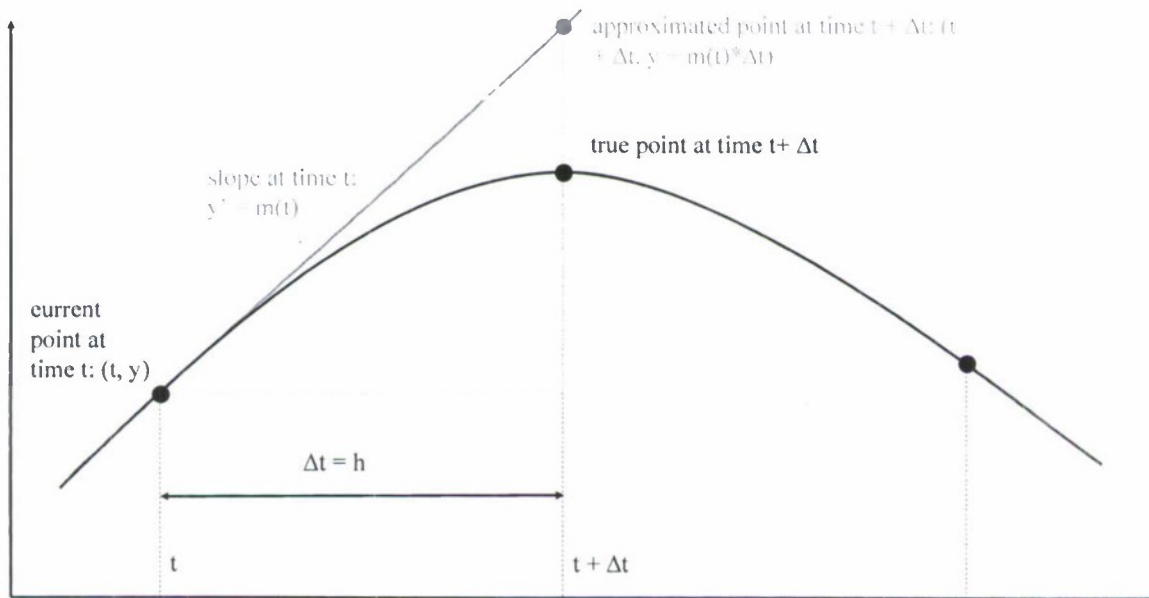
Intuitively, the second derivative (rate of change of the slope) of state variables could be employed to determine when to increase and decrease the time step. In areas with large second derivative, the time step should be small, in areas with low second derivative, the time step can be increased again. A first attempt to reduce the calculation expense is therefore to reduce the time step in the sections with high rate of change of the slope to become more accurate there. Then, when the rate of change of the slope becomes low again, the time step can be increased again. Such an approach was presented, e.g. in [Crouzet and Turinsky, 1996], and [Joukhadar and Laugier, 1996], and [Nairouz et al., 2009]. However, the approach in [Nairouz et al., 2009] was comparatively simple, as it was related to the specific model used, and did not provide a general approach towards the time stepping issue. Furthermore, it did not provide any means of error control. The resulting time steps were thus arbitrary in a sense that they could not be used in any other application, and did not give any information as to how far “off” the result was when the time step was varied, compared to a simulation with a very small fixed time step (which in turn would valid the assumption of correct results). Therefore, another approach needed to be found to provide a generally applicable algorithm for setting the co-simulation time step.

### 2.2.3 Description of numerical method for time step control of monolithic dynamic systems with known equations

The approach selected here for further investigation is taken from the numerical solution of ordinary differential equations (ODEs). In the very general case, dynamic systems are represented by ODEs. ODEs are equations of the form

$$y' = f(y, u, t) \quad (22)$$

This means that the derivative (= slope) of the underlying (generally unknown) base function is defined and computed through an equation that uses the current state  $y$ , an external input  $u$ , and the current system time  $t$ . These ODEs are used to model the dynamic systems in a computer. Given real time, these equations perfectly describe the dynamics of the modeled system. However, often such an equation cannot be solved, and hence the underlying base function needs to be approximated. Commonly this is done using digital computers. However, when using digital computers, the equation can only be solved stepwise in time since digital computers do not have infinite time resolution. This introduces an error to the approximation. The task therefore is to approximate the unknown underlying base function through a series of system time steps whose size must be determined in order to keep this error as low as possible, or within a prescribed range. Note that in the discussion for this report, the external input  $u$  is not considered. Also note that from here on out, the terms “slope” and “derivative” are used interchangeably.



**Figure 33: Euler's Method**

The very first and easiest approach to do this is using Euler's approach. Euler's approach takes the current position and time, and calculates a future point based on the resulting slope and given time step. With the slope  $y'$  of the current point  $y$ , and the prescribed time step  $h$  (here, the time step is denoted  $h$  instead of  $\Delta T$  or  $\Delta t$  to distinguish its

generality from the previous special case discussions), the next point is calculated using the equation

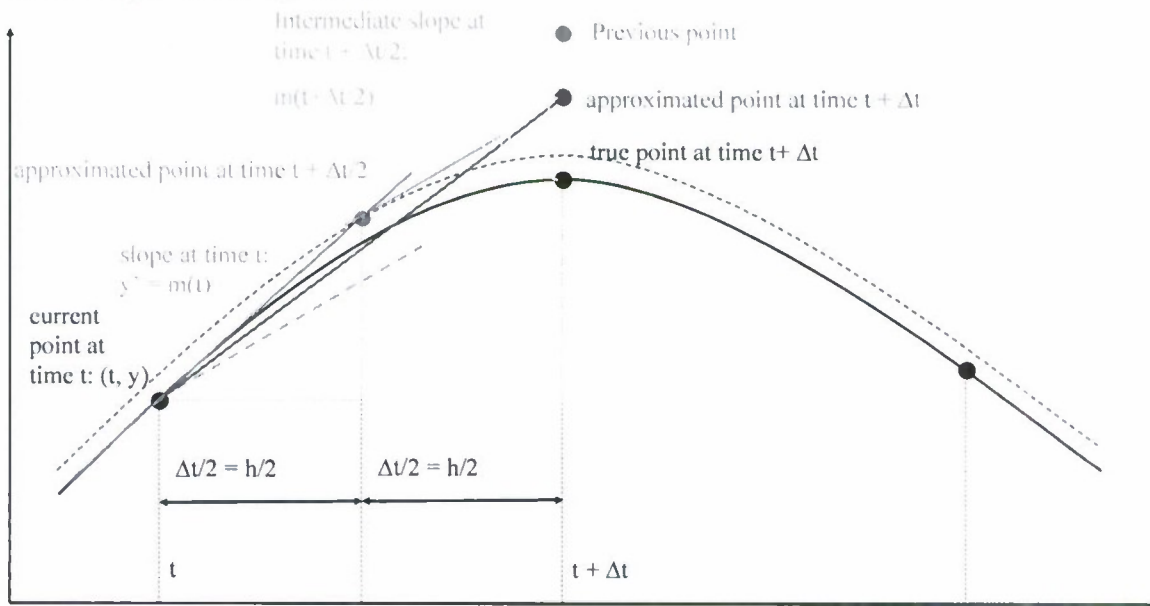
$$y_{new} = y_{current} + h * y' \quad (23)$$

See Figure 33 for explanation.

It can be readily observed that such an approach produces comparatively high errors. This is because it only takes into account the situation at the current point in time. If the slope of the underlying base function changes over the next time period with length  $h$ , then an error is introduced and it is in the order of magnitude of the time step  $h$ ,

$$\varepsilon \propto O(h) \quad (24)$$

There are two methods to reduce this error. The first one is to take some points in the future, calculate the slope there, and then apply a corrected slope at the current time step. This is depicted in Figure 34.



**Figure 34: Corrected Slope Approach (Heun's method)**

In this example, the slope at the next point in the future is used together with the slope at the current time step to calculate an average slope. This average slope is then used as the slope for the calculation for the next step. This approach is the second order Euler approach, also known as Heun's method. It can be seen that this method already represents a great improvement over the basic Euler method. The method can also be extended to go beyond just a single point in the future. By using more and more such points, the current slope can be approximated more and more precisely. It is clear that such an approach also increased the computational expense since the slope must be calculated at all the future points for each time step. A commonly used approach is to use four additional slopes for the algorithm, referred to as Runge-Kutta fourth-order method (RK4). It requires five function evaluations per time step, one for the current point and



four for the four future points. For an  $n$ -step Runge-Kutta method, the error is proportional to the time step taken to the power of  $n$ ,

$$\varepsilon \propto O(h^n) \quad (25)$$

The second method to reduce the error through the underlying base function approximation is clearly visible in Figure 33. The error will be reduced if the time step is reduced. Since the error is proportional to the time step, a reduction in time step by 50% will in general reduce the error by 50% as well. If this approach is used together with the slope approximation described above, the error can further be reduced. However, it must be kept in mind that reducing the time step will result in more points along the underlying base function to be calculated. So for both the described methods it holds that the decreased error comes along with an increased computational expense. It must also be noted that, while the above approaches do reduce the error, they do not provide for an error estimation or approximation. It can be seen that for different rates of change of the slopes, the error will be different. Hence, any method that future points at fixed time steps, and in general any method that uses fixed time steps, will not be useful for actually controlling the error. This can only be achieved through variable adaptive time steps.

Various such methods have been developed over the last decades, the most famous ones being of the Adams- and the Runge-Kutta-Fehlberg families. For this report and research, the Runge-Kutta-Fehlberg family of algorithms has been chosen. These algorithms were developed by Fehlberg [Fehlberg, 1969; and Fehlberg, 1970] on the basis of the Runge-Kutta algorithms described before. In general, these algorithms (like most other families of variable adaptive time step algorithms) are of the predictor-corrector type. They apply an  $n^{\text{th}}$ -order predictive step first, similar to the RK4 algorithms discussed above. However, instead of using the result of this step to approximate the next point, an additional future point is used as a corrector to cross-check the result obtained from the  $n^{\text{th}}$ -order step, thus rendering an  $(n+1)^{\text{th}}$ -order method. Based on the difference between the  $n^{\text{th}}$ -order and the  $(n+1)^{\text{th}}$ -order output, the error is estimated. This error estimation can then be used to either proceed with the approximation if the error is below a prescribed threshold, or to reduce the time step and repeat the approach at the current time step. More details on these algorithms can be found in the general literature, e.g. in [Burden and Faires, 2002]. For the method described in this report and research, the Runge-Kutta-Fehlberg-2-3 predictor-corrector (RKF23) method is employed. It creates a second order prediction and a third order correction, and was deemed a good compromise between accuracy and computational expense. The RKF23 algorithm is notionally described as follows (algorithm from [Ascher and Petzold, 1998]):

*User input:*

- starting system state  $y$  = current system state
- starting time step  $h$  = current time step
- prescribed error tolerance  $\tau$
- defined maximum time step  $h_{\max}$
- final simulation time  $T_{\text{final}}$

*begin*

```

let  $T = 0$ 
run until  $T \geq T_{final}$ 
    calculate state  $y$  and slope  $y' = k_1$  at current time step
    calculate slopes  $k_2$  and  $k_3$  at future points  $T+h/2$  and  $T+h$ , respectively
    let  $y_2 = y + h * k_2$ 
    let  $y_3 = y + h * (1/6*k_1 + 4/6*k_2 + 1/6*k_3)$ 
    let error  $\gamma = y_3 - y_2$ 
    if  $\gamma \leq \tau$  then
        let  $y_{new} = y_3$ 
        let  $T = T + h$ 
    set new time step  $h = \min(h_{max}, 0.8*h*(\tau/\gamma)^{0.25})$ 
end

```

Oftentimes, the dynamic model does not consist of only one single ODE, but rather of several ODEs that form a system of ODEs which describes the underlying system. The above algorithms can be adjusted to solve such system of ODEs, where each sub-system has its own time stepping approach, all systems are solved in parallel by one single solver. Applicable solvers for systems of ODEs have been developed for a long time, and are published in the general literature about dynamic simulation and numerical integration, see e.g. [Burden and Faires, 2002]. These algorithms have been continuously improved and modified for special cases [Mehrkanoon et al., 2009; and Savcenco and Mattheij, 2010]. In general, these solvers will take the different parts of the dynamic system, solve each one with its own variable adaptive time step, and synchronize the results at the end of the simulation time.

Solving a system of ODEs is a very similar problem to the co-simulation of dynamic systems. In essence, it is a method to approximate an unknown underlying base function. Both the underlying function for the ODE, and the underlying path of a state variable in a co-simulation, are unknown functions that need to be approximated. While the system of ODEs is generally solved by only one solver, in a co-simulation each sub-model has its own solver. Nevertheless, the models must be integrated and synchronized, just like in a system of ODEs. As described above, every sub-model has its own internal variable adaptive time step  $\Delta t$ . The simulation is carried out for every sub-model over a global time step  $\Delta T$ , after which the co-simulation is stopped, and the data is exchanged between the sub-models. Since the sub-models each have their own solver, whose internals usually can neither be observed nor changed, it must be assumed that the sub-models are executed with an internal time step  $\Delta t$  that is optimal as the solver sees it fit. Therefore, the initially proposed examination of internal vs. external time step is not relevant for the problem of co-simulation of different sub-systems. First, it can be assumed that the local time step  $\Delta t$  is chosen by each solver optimally for each time step. In the research proposed here, a different approach is chosen. Instead of using an internal time step  $\Delta t$  that is chosen by the solver, the internal time step  $\Delta t$  is set to be equal to the global time step  $\Delta T$ . This means that after every time step  $\Delta T$ , when the simulation is halted and the data are exchanged, and each sub-model has itself be solved over only one single time step. This need not be the case in every co-simulation, since as discussed above it is oftentimes not possible to impose such a constraint on the solver of each sub-model. But



it must be kept in mind that this is not relevant to the execution of the co-simulation itself. The described approach then leaves the global time step  $\Delta T$  as the time step to be varied and adapted to control the error of the co-simulation.

Due to the similarity of the problem, it is tempting to now claim that the techniques for the solution of a system of ODEs can be directly applied to a co-simulation environment. This would in theory be true if the system equations of the sub-systems were known. In particular, the slope of the state variables would have to be delivered as an output of the sub-models. However, this is oftentimes not the case. If for example precompiled legacy codes are used for sub-models, they may not give the derivatives of the state variables (i.e., the state variable slopes) as outputs. The same situation can arise if third party proprietary models are used. Even if the models are created by an in-house engineer using commercial simulation software, the outputs of the state variable slopes may not be available. For example, in the co-simulation environment created for the IRIS project, the fluid system was created with the commercial software Flowmaster. While the flow model was created by an in-house engineer, and thus its internals are known and understood, the software nevertheless does not provide state derivative outputs. In order to use the proven time step setting methods described above, the slope must therefore be acquired and approximated by some means before such algorithms can be applied.

#### **2.2.4 Application of the adaptive time step method to the double pendulum sample problem**

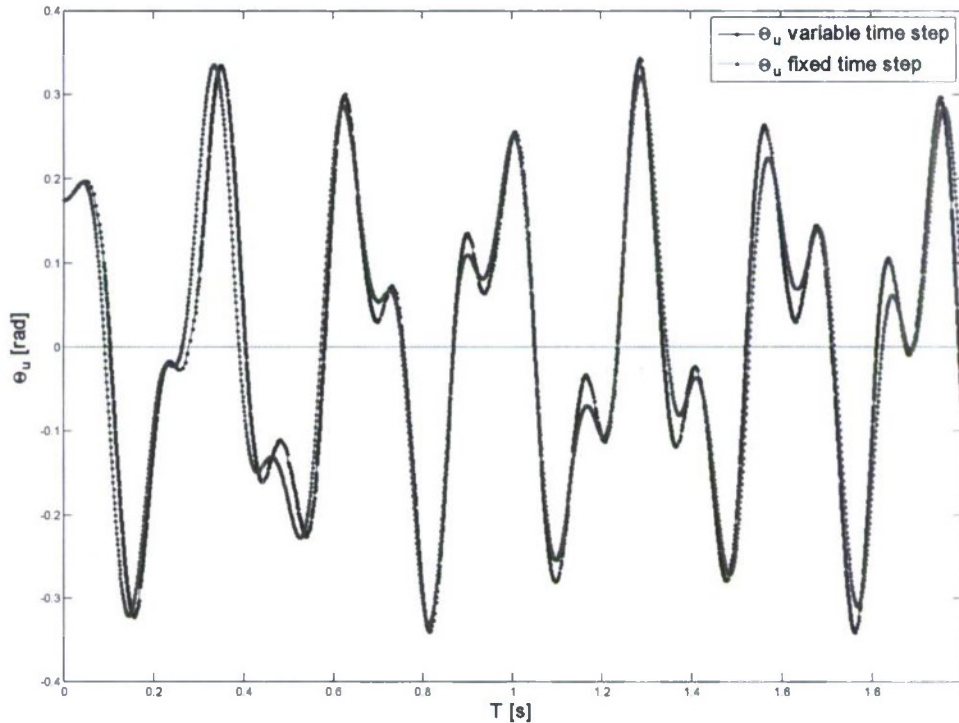
The split and integrated double pendulum model that was described in detail above is used to implement the proposed RKF23 into a co-simulation environment with “unknown” models and without the state derivatives available (this is not theoretically correct, however, since the state derivatives indeed are readily available from the sub-models; they are however, assumed to be unknown, and used only for evaluation). As mentioned, the sub-model local time steps  $\Delta t$  are equal to the global time simulation time step  $\Delta T$ , at each of which the simulation stops and exchanges variables.

The main problem in this setup, and in the research proposed here, is that the state derivatives (the slopes) are unknown. In order to be able to implement the proven time stepping algorithms from numerical simulation, these derivatives must be acquired using the available state data information. In general, this might turn out to be an involved process. This falls into the area of scattered data approximation, which has been discussed in many publications, for example in [Wendland, 2005]. In the double pendulum model, this is achieved by using the current point, the previous two points, and the future steps at  $T+h/2$  and  $T+h$  (see notional algorithm listing above) to construct a four-point third-order polynomial, whose derivatives are then taken at the current and future points. The initial use of a second-order three-point polynomial has proven to not work with the RKF23 algorithm, because due to its nature, the second derivative for such a polynomial is equal at all three points. From the notional algorithm listing above it can be seen that such behavior leads to a predictor-corrector error  $\gamma$  that is zero at all times. Hence, the algorithm sees no reason to change the time step, and it will use the same initial time step during the whole simulation run. The four-point polynomial is constructed using the approach described in [Vanderplaats, 2005]. The values of  $y$ ,  $y_2$ ,



and  $y_3$  in the above algorithm are obtained by running the simulation over the necessary time ranges  $h/2$  and  $h$ , starting from the current position.

Figure 35 depicts the result for the upper pendulum bar of the double pendulum.

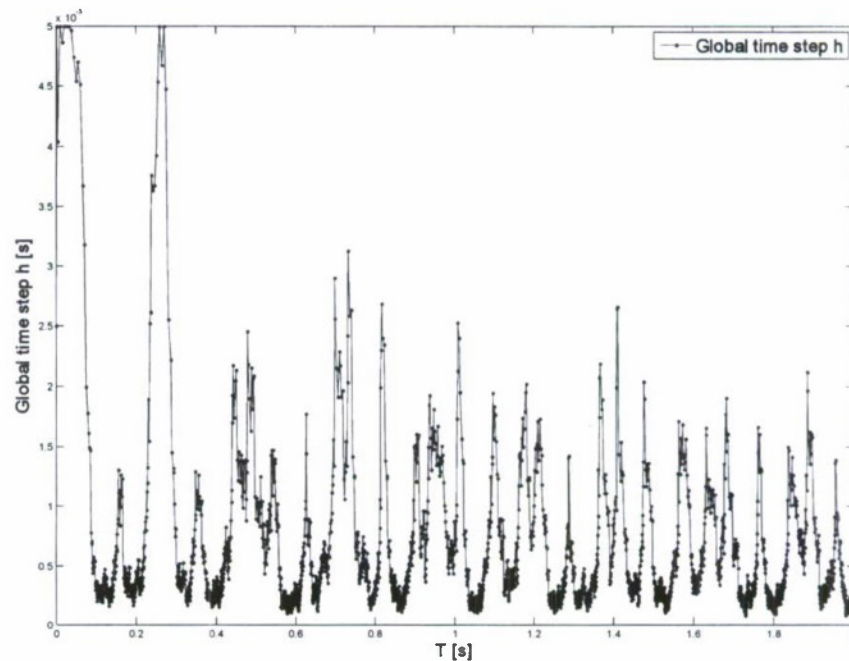


**Figure 35: Double Pendulum Simulation Result, Upper Pendulum Deflection Angle**

The red curve shows the actual pendulum deflection angle  $\Theta_1$  when the time step is fixed and very small. The blue line shows the result for the applied time stepping algorithm. In general, the agreement between the curves is acceptable. However, especially in the regions where the rate of change of the slope is high, the time stepping algorithm tends to overshoot the ideal result.

Figure 36 shows how the time stepping algorithm varied the time step over the run time of the simulation.

It can be observed that there are regions of strong fluctuations where the derivatives reach their maximum and minimum. The reason for this behavior is as of now not understood and subject to intense research. The result of this behavior is that due to these fluctuations, the deviation error  $\gamma$  (as per notional algorithm description) in these regions becomes very high, resulting in repeated simulation runs and very small time steps where it is not necessary. This behavior furthermore impacts the simulation result at the turning points of the pendulum deflection angle graph. Here, the time step should ideally be lower but is negatively impacted by the behavior of the calculated slopes. In order to compare the calculated slopes with the “correct” slopes available from the sub-models themselves, the



**Figure 36: Time Step Settings of Double Pendulum Simulation**

deviations of these slopes were calculated. See Figure 38 for a comparison. It can be observed that there is strong chattering in the deviations, indicating that the calculated slopes deviate from the “correct” slopes strongly and with large variability in certain regions. These variations directly impact the results of the time stepping algorithm, rendering it much less effective as it should in theory be. Hence, there needs to be further investigation into why this chattering happens, and what measure can be taken in order to reduce or completely avoid the fluctuations. An investigation into the stability of the simulation against randomly set time steps that are not a function of the current slope has shown that the simulation is very resistant against time step fluctuations themselves. This indicates that the implementation of the time stepping algorithm itself incurs some behavior into the simulation that leads to the chattering of the slopes and the subsequent time stepping behavior. A preliminary analysis indicates that it might be the time step itself that leads to the deviations. If the time step is very small, even slight deviations of the state variables lead to large changes in state derivatives between two subsequent time steps. Since a simulation is never as smooth as an ODE, this might be the cause of the strong derivative variations. If this is the cause, then a possible remedy could be to use curve smoothing, or to implement a minimum time step in order to decrease the possible magnitude of the derivative jumps.

Figure 37 shows the calculated state derivatives (slopes)  $k_1$ ,  $k_2$ , and  $k_3$  over the length of the simulation run.



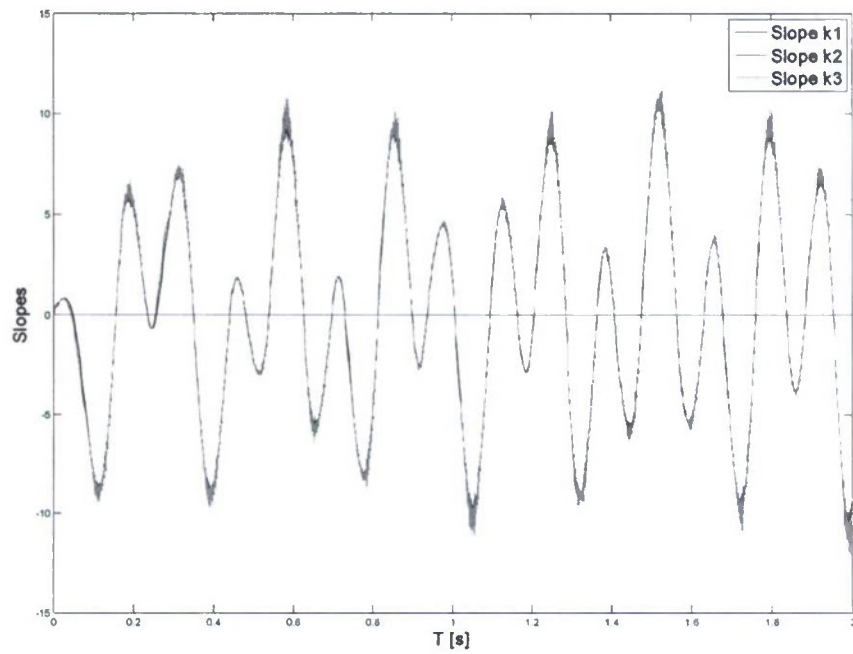


Figure 37: Slopes k1, k2, and k3 for Double Pendulum Simulation

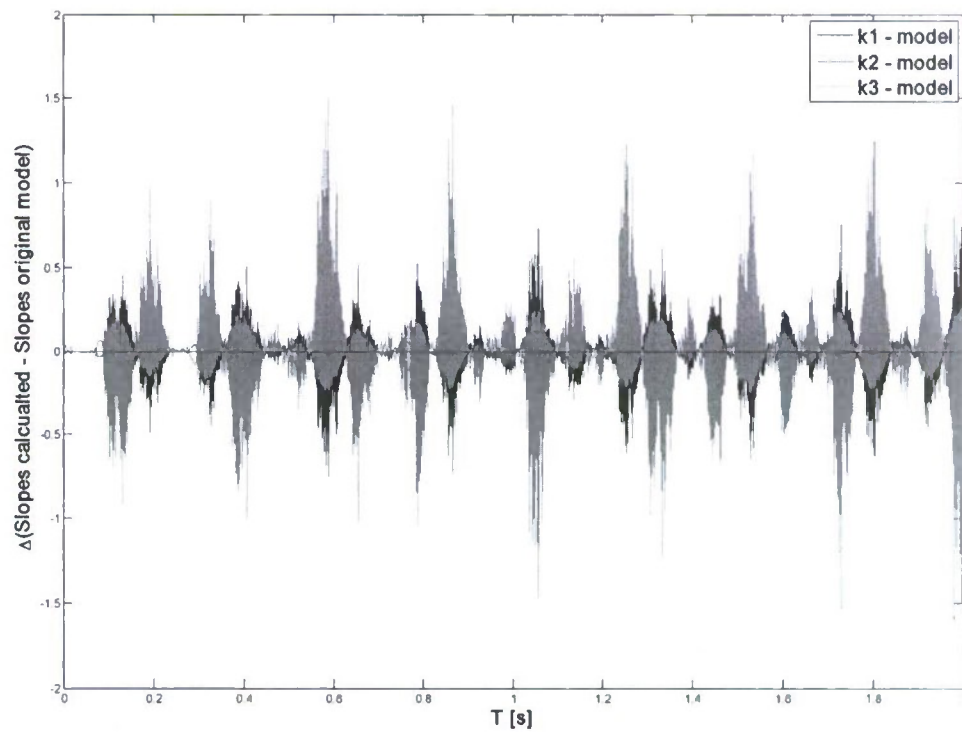


Figure 38: Deviations of Calculated and “Correct” Slopes

## **2.3 Discussion of approach with respect to proposed subtasks**

In the previous report, three subtasks have been proposed:

1. Determination of real-world representation of co-simulated system
2. Definition of optimal ratio between global and local time steps
3. Definition of corrective actions for accurate co-simulation execution

Subtasks 1 and 3 are handled through the application of a proven algorithm from the numerical integration of ODEs. The conditions for these algorithms are clear and known. The algorithms themselves are mathematically proven and rigorous. They are employed to approximate an unknown underlying function, or “path” that a variable has to take. The underlying problem is the same for both applications “solving an ODE” and “obtaining the path of a co-simulation state variable”. Therefore, the application of such an algorithm to a co-simulation handles the issues in subtasks 1 and 3, provided that the simulation itself is set up properly (which is an underlying assumption in this research, and a given condition for the double pendulum model used).

Subtask 2 is itself not an issue for research. It has been discussed above, that the application of an “optimal” ratio between local and global time steps is not critical to the execution of the co-simulation. In the case where the sub-model solver is unknown, this solver will choose an internal time step that it deems “optimal” to solve the model within the given global time step. Its internal local time step is neither controllable nor accessible, and hence it cannot be changed. It must be assumed to provide the best solution to the sub-model under consideration. In the case where the local time step can be accessed and modified through the simulation, it can either be set by implementing an algorithm from numerical integration, or it can be set to “1” which represents one single time step from beginning to end of one global time step execution. By implementing the latter approach, the simulation engineer obtains full time step control over the sub-model. This is the best scenario for an adaptive time step as discussed here.

## **2.4 Results discussion and future work**

The proposed approach for time stepping using an RKF23 algorithm has been discussed and implemented on a simple example problem. For this research, it is assumed that the co-simulation is set up and working correctly. The general issues relating to dynamic system M&S and co-simulation itself (such as algebraic loops, inter-model constraints, stability, etc.) have been studied for decades and are ongoing subjects in research and literature, but not subject of the research presented in this report.

The algorithm has been shown to be a promising way of achieving a mathematically reasoned time step setting for co-simulation. It must be understood though, that this research is currently only at the beginning. It has been shown that the results, while acceptable, are not yet satisfactory. Also, once implemented properly for the simple example problem, the algorithm’s validity for general problems needs to be evaluated.

The current main problem is the obtainment of the slopes of the state variables. It has been shown that there are cases of strong chattering. Since the simulation itself is



resistant against variable time steps, the time stepping algorithm itself is the cause of the deviations. The chattering and general behavior of the state derivatives is therefore subject to further investigations, and currently one of the main focal points. Some of the points to be taken into account have been discussed above.

If the time stepping algorithm is implemented successfully, it must be extended to take into account different sub-model frequencies. Currently, the application is such that the simple example problem is composed of two sub-models with similar system dynamics. If however, there are two or more dynamic systems with different system dynamics integrated into a co-simulation, a time stepping algorithm should ideally be able to cope with this situation. Like in the solution of a set of ODEs (as discussed above), the algorithm needs to take into consideration the time steps necessary for each of the different dynamic systems. This will require a more sophisticated time stepping schedule. In particular, if there exist systems with slow dynamics and systems with fast dynamics, the fast systems might require far smaller time steps than the slow systems. The algorithm therefore needs to take into consideration that such deviations exist. Slower sub-models could be executed far less often than fast sub-systems. The missing data for the fast systems might be obtained through means of interpolation. Such an algorithm would greatly improve the co-simulation by significantly reducing the necessary computational load.

Also an issue is the selection of a maximum time step ( $h_{\max}$  in the notional RKF23 algorithm). This is important because the algorithm itself might choose time steps that are reasonable with respect to the time stepping routine itself, but not acceptable for a dynamic simulation. This would be the case for example when a dynamic system goes through a phase of little to no change in state variable outputs. Subsequently, the algorithm would choose a higher and higher time step. If the simulation then comes across a steep change in system states, the algorithm would still be at too high a time step to properly capture such an event. The results would then become out of control. This behavior was also observed in the simple example double pendulum problem. By choosing a maximum time step, and possibly implement a rollback algorithm, such problems can be prevented. However, the issue is how to find the maximum time step. In this example, methods from system analysis can be employed; however, this will be another main point of research focus, as system analysis has its drawbacks and is not generally applicable.

## **Task 3: Surrogate Modeling of Fluid Cooling Systems**

### ***3.1 Introduction***

The proposed tasks for this research include the implementation of the method of component surrogate modeling with graph-based integration to the fluid system model built in the DOMINO environment (denoted as “the DOMINO model” below), based on the successful development of the method in the early stage of the research. The objective of this research was to demonstrate the applicability of the developed surrogate modeling method by implementing it to an actual naval fluid system design problem.



Although the graph-based surrogate modeling method was developed, tested, and demonstrated successfully using the notional YP fluid model, it was aware that the change of the original plan for implementing the method by using the DOMINO model was inevitable. This is because the use of the DOMINO model for this research might be restricted by the export control regulations. As an alternative to the DOMINO model, the CW-RSAD model was chosen to perform the proposed research. There are two main reasons for choosing the CW-RSAD model as the proof of implementation: the CW-RSAD model has been broadly used as a concept of demonstration in various research efforts supported by the U.S. Navy; and the system configuration of the CW-RSAD model seems complicated enough to reflect the real-world problem – or, at least, as complicated as the DOMINO model. In addition, the use of CW-RSAD model as the demonstrator reveals one of the advantages exhibited by the component surrogate modeling method with graph-based integration, which is the model reusability. The CW-RSAD model is the one from which the notional YP fluid model originated so the component surrogate models generated for the notional YP fluid model can be reused without modification for creating the CW-RSAD surrogate model.

In this section, the formulation of the method of component surrogate modeling with graph-based integration will be introduced and its implementation to the damage analysis of the notional YP fluid system will be discussed. Python, an object-oriented scripting language, is chosen as the development platform for this implementation. Then, the application of the method to a larger system, the CW-RSAD model, will be conducted and the results will be presented and discussed.

## **3.2 Research Details**

### **3.2.1 Component Surrogate Modeling with Graph-Based Integration**

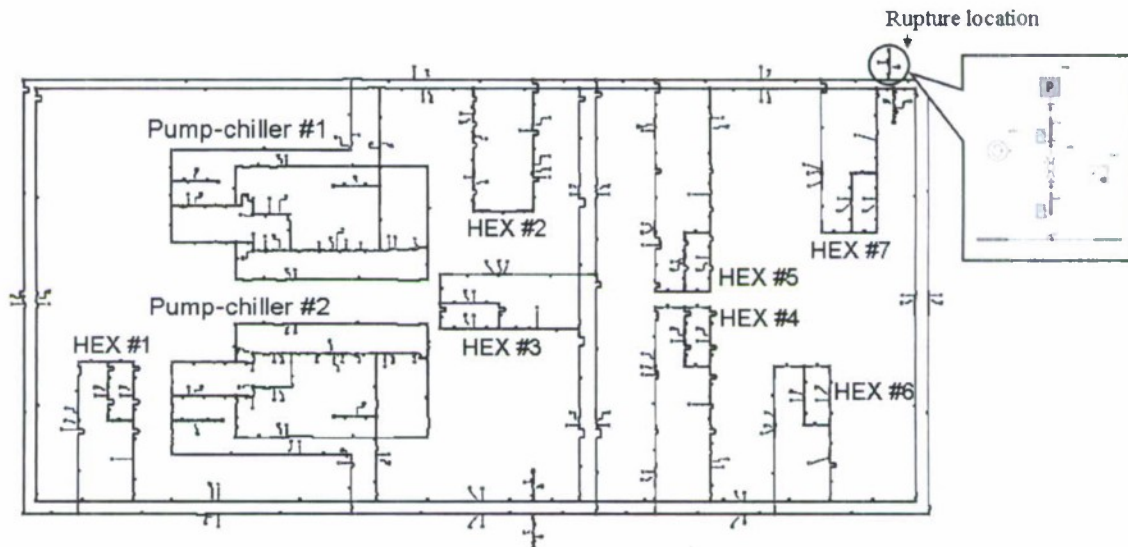
This surrogate modeling method is capable of providing a flow-based engineering systems M&S environment that can treat the connection-topological configuration the network as a “design variable” or “noise variable” in design space exploration, design trade study, or design optimization process. As a result, this method will also enable a simulation-based design for resilience and survivability against propagating failures of the system.

The method encompasses two key ideas: one is the surrogate modeling of the object-oriented component models, and the other is the graph-based topological modeling that connects all the component surrogate models and manages the changes of the connections among them. As mentioned previously, the method will be described very briefly and demonstrated using part of the implementation of the notional YP fluid system. The comprehensive details of the method and its implementation to the notional YP fluid system can be found in [Moon, 2010].

#### **3.2.1.1 Notional YP Fluid System Model**

Figure 39 depicts the Flowmaster V7 model of the notional YP fluid system. The notional YP fluid model consists of a pair of redundant pump-chiller sub-networks, seven heat exchanger units, and six thermal service loads. In the model, the heat exchanger units

HEX no. 4 and HEX no. 5 are redundant heat exchanges serving a single thermal load. Each heat exchanger contains either one or two (as redundancy) flow control valve(s). In this example, the system has 18 damage control valves, 11 flow control valves, and 10 valves for load isolation.



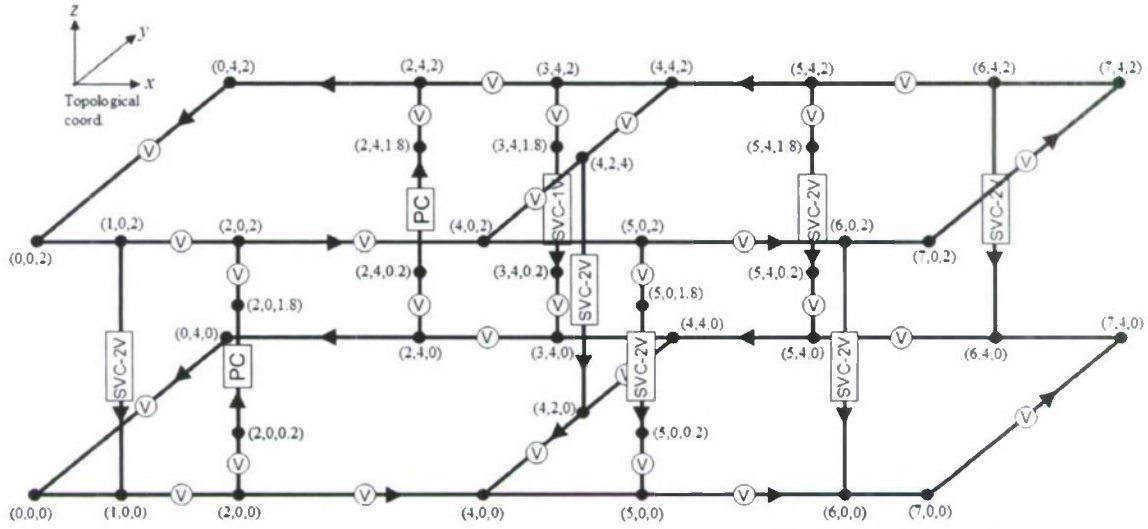
**Figure 39: Flowmaster V7 Model of Notional YP-Fluid System (HEX: Heat Exchanger Unit)**

### **3.2.1.2 Definition of Components and Graph Representation**

The first step is to translate the notional-YP fluid model in Figure 39 into a graph representation that is composed of edges and nodes, such as the one shown in Figure 40. An interesting aspect of the graph model in Figure 40 is that it has 52 edges and 40 nodes; however, all these graph components can be grouped into five different component types. A component type can be thought of as a class in object-oriented programming for generating multiple edge component instances with different values of certain properties. That being said, all the edge components can be represented by five component models that allow a few model properties, or model parameters, to be variable which can improve the model's reusability. Table 3 is the list of the component models defined in the graph-based model representation.

### **3.2.1.3 Generation of Component Surrogate Models**

The mathematical models of five component types listed in Table 3 are then generated using the recurrent neural network (RNN)-based surrogate modeling method [Moon, 2010]. In this particular implementation, eight neural-net (NN) surrogate models were created to model the five different types of components. **Table 4** shows the specification of the NN-surrogate models.



**Figure 40: Graph Representation of Notional YP Fluid Model**  
(Circles with a “V” indicates the location of smart valves for automated damage control)

**Table 3: Edge Component Types**

| Name   | Component type           | No. of comps | State vars                 | Boundary cond.          | Control vars                         | Parameters                   |
|--------|--------------------------|--------------|----------------------------|-------------------------|--------------------------------------|------------------------------|
| PIPE   | Simple pipe              | 14           | $q \text{ (m}^3/\text{s)}$ | $\Delta P \text{ (Pa)}$ | None                                 | $l \text{ (length, m)}$      |
| VPIPE  | Pipe with a valve        | 30           | $q$                        | $\Delta P$              | $v_1 \text{ (0 to 1)}$               | $l, d \text{ (diameter, m)}$ |
| SVC-1V | Sve load with a valve    | 1            | $q$                        | $\Delta P$              | $v_1$                                | None                         |
| SVC-2V | Sve load with two valves | 6            | $q$                        | $\Delta P$              | $v_1, v_2$                           | None                         |
| PC     | Pump-chiller sub-net     | 2            | $q_{in}, q_{out}$          | $P_{in}, P_{out}$       | $v_1, \omega_{pump} \text{ (rad/s)}$ | None                         |

**Table 4: NN-Surrogate Model Specifications**

| Name   | NN-functions   | Input range   | Data size | No. of neurons | Training MSE            | Test MSE                |
|--------|----------------|---|-----------|----------------|-------------------------|-------------------------|
| PIPE   | pipe           | $-0.001 \leq q \leq 0.001$<br>$-10^4 \leq \Delta P \leq 10^4$<br>$0.5 \leq l \leq 5$  | 21,511    | 6              | $1.1351 \times 10^{-4}$ | $3.1619 \times 10^{-4}$ |
| VPIPE  | vpipe_0127     | $d = 0.0127$<br>$0.3 \leq l \leq 1.5$<br>$-0.001 \leq q \leq 0.001$<br>$-10^5 \leq \Delta P \leq 10^5$  | 37,817    | 8              | $8.3003 \times 10^{-4}$ | $1.7213 \times 10^{-3}$ |
|        | vpipe_01905    | $d = 0.01905$<br>$0.3 \leq l \leq 6.5$<br>$0 \leq v_1 \leq 1$   | 102,285   | 8              | $1.9184 \times 10^{-3}$ | $2.9164 \times 10^{-3}$ |
|        | vpipe_0254_slp | $d = 0.0254$<br>$2 \leq l \leq 7$<br>$-0.001 \leq q \leq 0.001$<br>$-4 \times 10^3 \leq \Delta P \leq 3 \times 10^3$  | 26,694    | 8              | $1.0910 \times 10^{-5}$ | $4.9561 \times 10^{-5}$ |
|        | vpipe_0254     | $0 \leq v_1 \leq 1$<br>$-0.0005 \leq q \leq 0.0005$<br>$-7 \times 10^4 \leq \Delta P \leq 7 \times 10^4$  | 29,271    | 16             | $2.0953 \times 10^{-5}$ | $3.1013 \times 10^{-3}$ |
| SVC-1v | sve_1v         | $-0.0005 \leq q \leq 0.0005$<br>$-1 \times 10^5 \leq \Delta P \leq 1 \times 10^5$<br>$0 \leq v_1 \leq 1$  | 10,077    | 7              | $1.1351 \times 10^{-4}$ | $3.1619 \times 10^{-4}$ |
| SVC-2v | sve_2v         | $-0.0003 \leq q \leq 0.0003$<br>$-5 \times 10^4 \leq \Delta P \leq 1 \times 10^5$<br>$0 \leq v_1 \leq 1$  | 10,077    | 11             | $1.8831 \times 10^{-5}$ | $4.5144 \times 10^{-5}$ |
| PC     | pc             | $-0.001 \leq q_{in}, q_{out} \leq 0.002$<br>$10^5 \leq P_{in} \leq 2.5 \times 10^5, 10^5 \leq P_{out} \leq 4.5 \times 10^5$<br>$0 \leq v_1 \leq 1, 0 \leq \omega_{pump} \leq 400$ | 13,200    | 11             | $5.7994 \times 10^{-4}$ | $9.3825 \times 10^{-4}$ |



The YP fluid model is a scaled-down version of the CW-RSAD model, which indicates that all these NN component surrogate models can also be used for modeling the CW-RSAD model.

#### **3.2.1.4 Component Model Integration Using Incidence Matrix of Graph Model and Simulation**

The strength of a graph model reveals when its various matrix forms are incorporated with numerical applications. One of the most frequently used matrix form of graph models may be the incidence matrix, which describes the connectivity of the network system.

In the developed method, the incidence matrix of the graph model shown in Figure 40 is used as the “glue” of the edge component models. In simulation, the numerical solver algorithm uses the incidence matrix of the graph model and the flow rate outputted from the edge component models to generate the algebraic equations of the Kirchhoff's Current Law (KCL) with the flow continuity constraints imposed by the system connection topology. Then, the solver finds, at each simulation time step, the pressure values at the nodes (i.e., the connection points of the edge component models) that satisfy the KCL constraints of the system. In this setting, multiple models with different connection topologies are generated with different incidence matrices, while still utilizing the same pool of component surrogate models.

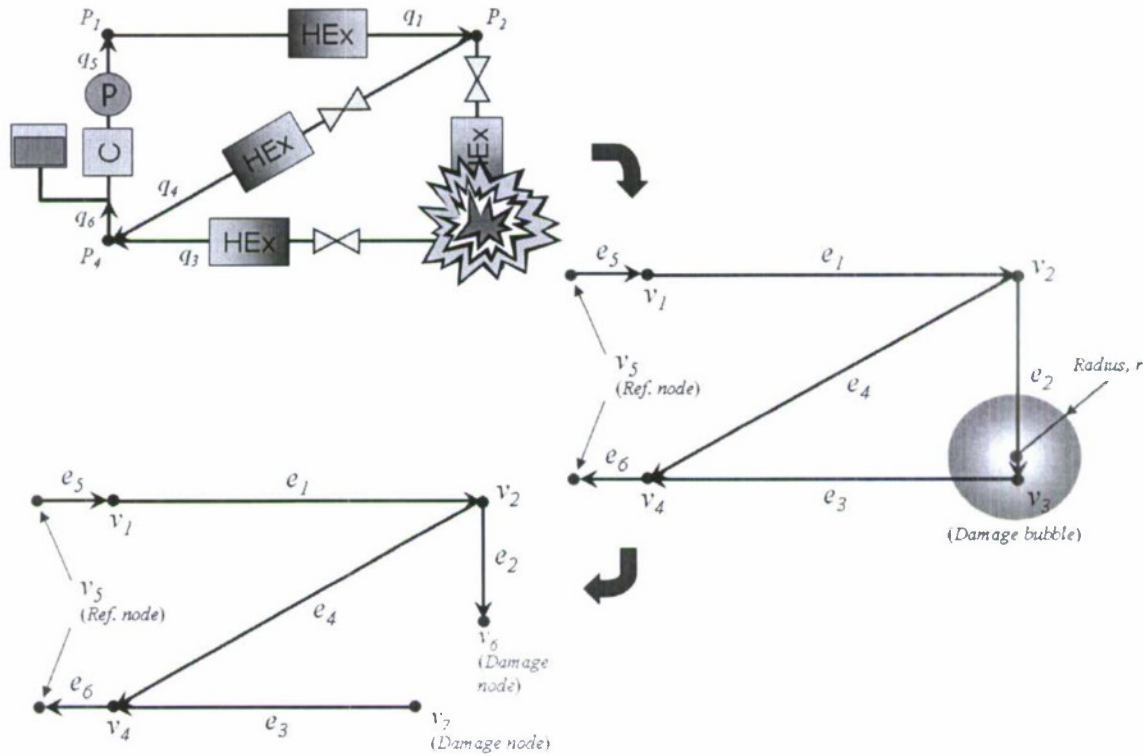
#### **3.2.1.5 Damage Modeling**

For the flow system model based on the developed method, the damage in the flow system can be modeled by combining the connection topological changes of the model and the change in the properties of the component models resulted from the damage. Basically, the connection topological changes by damage are realized in the same way as for creating the model with a different connection topology – by modifying or changing the incidence matrix of the model, which makes modeling damage very easier than many other modeling approaches. Since the damage often causes the changes of the physical properties, such as the pipe lengths of the affected edge component models, the properties of the affected models are modified too. Figure 41 briefly describes how the damage is implemented to a simple water-based cooling system which consists of a single chiller, four heat exchangers, and three valves. In Figure 41, an explosion happens to the system, and the area that is influenced by the explosion is represented by a “damage bubble,” a sphere with the radius  $r$ . This simple damage modeling approach assumes that every part of the components that are inside this bubble is completely destroyed, so the simulation generates a new model topology and changes the component model properties according to the given damage, as shown in the model at the lower-left corner of Figure 41.

#### **3.2.1.6 Auto-Generation of the Damage Control System Model**

A damage analysis of a fluid network system without any damage control effort is meaningless, since the analysis would yield only a trivial solution of the total system failure by eventually shedding all the reserved fluid out of the system. In the notional YP fluid model and the CW-RSAD model, the automated damage isolation is done by the cooperative and reactive actions of the smart valves that equipped in the fluid network

system. The graph-based surrogate modeling method provides an algorithm for auto-generation of a distributed damage control system model for the given model. This damage control model and auto-generation algorithm is particularly useful for performing simulation-based design for resilience of intelligent and reconfigurable fluid systems by conducting the damage analysis in the early design stage.



**Figure 41: Damage Modeling in Simple Fluid System**  
(C: Water Chiller, P: Pump, HEX: Heat Exchanger)

### **3.2.1.7 Simulation Settings**

For both the notional YP fluid model and the CW-RSAD model, the simulation ran with a discrete time step of 0.05 second, and throughout the whole simulation. The PC no.1 (pump-chiller sub-network, see Figure 39) operated with a single pump turned on to a fixed speed of 200 rad/s (about 1910 RPM), while PC no.2 was off. When PC no.1 was damaged, the controller on PC no.2 turned the pump of the PC no.2 on in order to continue the system-level operation. For all simulation runs, the damage bubble representing the damage was set to a constant radius, which was 0.7.

### **3.2.1.8 Comparison of Simulation Performance**

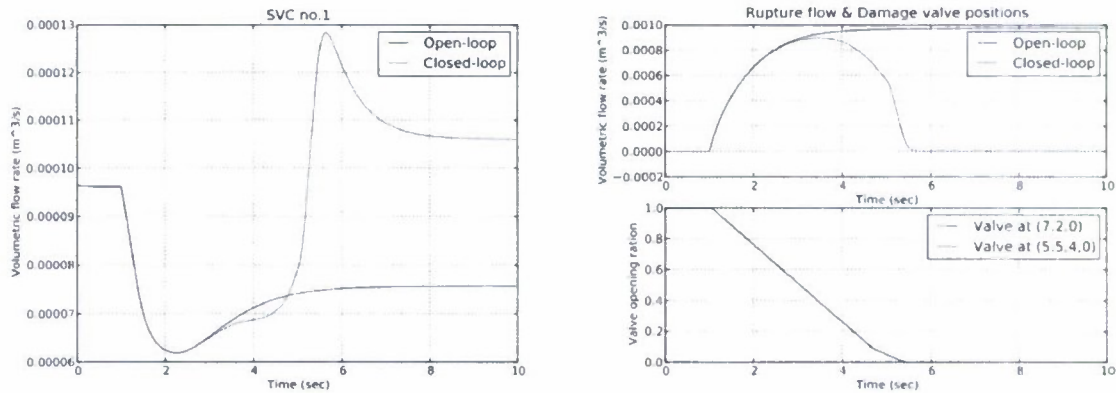
The analysis was performed with the simulation running for 5 seconds of simulation time, the total wall-clock time taken by the simulation with the original Flowmaster model and the graph-based surrogate model of the notional YP fluid system was measured. The result of this particular comparison test showed that the graph-based surrogate model



took 4.48 secs which was 12.6 times faster than the Flowmaster model, which took 56.48 secs. However, on the other side, the graph-based surrogate model contained errors and biased information in its response based on the responses of the Flowmaster model. This trait is in fact common to every surrogate model due to the fact that the surrogate model is an approximation of a model. Despite of such a problem, the graph-based surrogate model would be very useful, especially for conduct analysis in the early phase of the design process. In practice, the Flowmaster model is not necessarily more accurate than the graph-based surrogate model when it comes to the early design stage because a large portion of the system details is yet unknown or not decided at all. Instead, what is needed more in the early design stage is a computationally affordable model that allows a designer to run a large number of analysis cases for exploring as large a design space as possible, meaning that the graph-based model is an effective choice to meet such requirements.

### 3.2.1.9 Damage Simulation and Analysis

In this analysis, the system was closed by the reference damage control, and 28 simulations are compared, each of which has a different damage location. Figure 42 shows the comparison of the open-loop and the closed-loop responses of the graph-based surrogate model with the same damage condition. In Figure 42(a), right after the rupture occurred at one second point of the simulation, the closed-loop system had a similar sudden drop of flow rate as the open-loop system, but eventually settled into a new recovered steady state. Although Figure 42 shows only one of the 28 results with the different damage conditions, the behaviors of other cases should be similar to Figure 42.



(a) Flow Rate at SVC no.1 (b) Total Rupture Flow and Valve Inputs  
**Figure 42: Comparison of Open-Loop and Closed-Loop Responses of YP-Fluid Model with Rupture at (7, 4, 0)**

In order to quantify and measure the system recovery performance for every simulation case, the operation capability rate (OCR) was defined and given by Equation (26):

$$OCR = \frac{w_1 \tilde{q}_1^f + w_2 \tilde{q}_2^f + w_3 \tilde{q}_3^f + w_4 \cdot \max(\tilde{q}_4^f, \tilde{q}_5^f) + w_5 \tilde{q}_6^f + w_6 \tilde{q}_7^f}{w_1 q_1^o + w_2 q_2^o + w_3 q_3^o + w_4 \cdot \max(q_4^o, q_5^o) + w_5 q_6^o + w_6 q_7^o} \quad (26)$$

and,



$$\tilde{q}_i^f = \begin{cases} q_i^f, & \text{if } q_i^f \leq q_i^o \\ q_i^o, & \text{otherwise} \end{cases} \quad (27)$$

where,  $q_i^o$  and  $q_i^f$  ( $i = 1, \dots, 7$ ) are the initial and the final values of the volumetric flow rate, respectively, at the 7 heat exchanger units, and  $w_j$  ( $j = 1, \dots, 6$ ) are the weight coefficients for the six service loads in the system. In Equation (26), the terms  $w_4 \cdot \max(q_4^o, q_5^o)$  and  $w_4 \cdot \max(\tilde{q}_4^f, \tilde{q}_5^f)$  reflected that the two heat exchangers SVC no.4 and SVC no.5 are redundant.

The OCR is in a scale of 0 to 1 which represents a measure of how well the recovered system held its chilled-water delivery capacity from the level of the system before damage. Given the formulation of Equation (26), the OCR strongly relies on the right choice of the weight coefficients which represent the service load priorities based on customer requirements, mission profile, and design philosophy. In this example, they were chosen by the author for demonstration purpose and given in Table 5.

A system's recovery performance should not be measured only using the final system state but also by how fast the system recovered. The OCR in Equation (26) was not formulated in this way just because the model with the reference control model has no control-induced delay in damage control efforts. However, in the real application, the recovery speed must be taken into account in the formulation of OCR.

Figure 43 presents the result of the damage analysis resulting from the 28 damage cases.

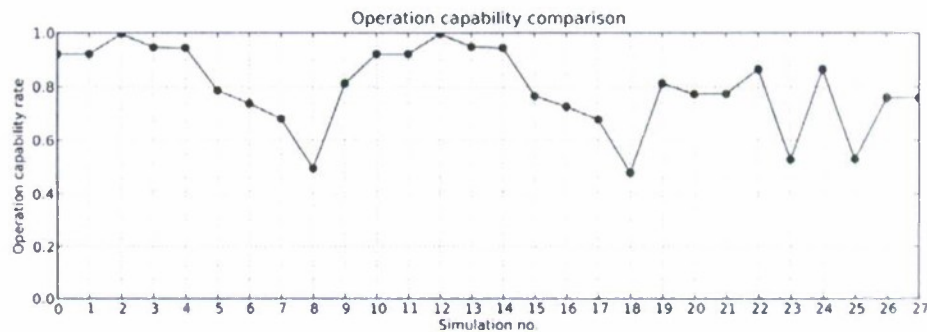


Figure 43: Operation Capability Rates for Notional YP Fluid Model

Table 5: Component Failure Status for Notional YP Fluid Model

| Simulation no. | Weight (1-10) | Component | Rupture location |      |      |      |      |      |      |            |      |      |      |      |      |      |                 |      |      |      |      |      |      |            |      |      |      |      |      |      |
|----------------|---------------|-----------|------------------|------|------|------|------|------|------|------------|------|------|------|------|------|------|-----------------|------|------|------|------|------|------|------------|------|------|------|------|------|------|
|                |               |           | Starboard lower  |      |      |      |      |      |      | Port lower |      |      |      |      |      |      | Starboard upper |      |      |      |      |      |      | Port upper |      |      |      |      |      |      |
| 0              |               |           | 0                | 1    | 2    | 3    | 4    | 5    | 6    | 7          | 8    | 9    | 10   | 11   | 12   | 13   | 14              | 15   | 16   | 17   | 18   | 19   | 20   | 21         | 22   | 23   | 24   | 25   | 26   | 27   |
| 1              | PC #1         |           | 0                | 0    | 0    | 0    | 0    | 0    | 0    | 0          | 0    | 0    | 0    | 0    | 0    | 0    | 0               | 0    | 0    | 0    | 0    | 0    | 0    | 0          | 0    | 0    | 0    | 0    | 0    | 0    |
| 2              | PC #2         |           | 0                | 0    | 0    | 0    | 0    | 0    | 0    | 0          | 0    | 0    | 0    | 0    | 0    | 0    | 0               | 0    | 0    | 0    | 0    | 0    | 0    | 0          | 0    | 0    | 0    | 0    | 0    | 0    |
| 3              | HEX #1        |           | 0                | 0    | 0    | 0    | 0    | 0    | 0    | 0          | 0    | 0    | 0    | 0    | 0    | 0    | 0               | 0    | 0    | 0    | 0    | 0    | 0    | 0          | 0    | 0    | 0    | 0    | 0    | 0    |
| 4              | HEX #2        |           | 0                | 0    | 0    | 0    | 0    | 0    | 0    | 0          | 0    | 0    | 0    | 0    | 0    | 0    | 0               | 0    | 0    | 0    | 0    | 0    | 0    | 0          | 0    | 0    | 0    | 0    | 0    | 0    |
| 5              | HEX #3        |           | 0                | 0    | 0    | 0    | 0    | 0    | 0    | 0          | 0    | 0    | 0    | 0    | 0    | 0    | 0               | 0    | 0    | 0    | 0    | 0    | 0    | 0          | 0    | 0    | 0    | 0    | 0    | 0    |
| 6              | HEX #4        |           | 0                | 0    | 0    | 0    | 0    | 0    | 0    | 0          | 0    | 0    | 0    | 0    | 0    | 0    | 0               | 0    | 0    | 0    | 0    | 0    | 0    | 0          | 0    | 0    | 0    | 0    | 0    | 0    |
| 7              | HEX #5        |           | 0                | 0    | 0    | 0    | 0    | 0    | 0    | 0          | 0    | 0    | 0    | 0    | 0    | 0    | 0               | 0    | 0    | 0    | 0    | 0    | 0    | 0          | 0    | 0    | 0    | 0    | 0    | 0    |
| 8              | HEX #6        |           | 0                | 0    | 0    | 0    | 0    | 0    | 0    | 0          | 0    | 0    | 0    | 0    | 0    | 0    | 0               | 0    | 0    | 0    | 0    | 0    | 0    | 0          | 0    | 0    | 0    | 0    | 0    | 0    |
| 9              | HEX #7        |           | 0                | 0    | 0    | 0    | 0    | 0    | 0    | 0          | 0    | 0    | 0    | 0    | 0    | 0    | 0               | 0    | 0    | 0    | 0    | 0    | 0    | 0          | 0    | 0    | 0    | 0    | 0    | 0    |
| 10             | OCR           |           | 0.92             | 0.92 | 0.99 | 0.89 | 0.99 | 0.82 | 0.75 | 0.74       | 0.55 | 0.81 | 0.92 | 0.92 | 0.99 | 0.89 | 0.89            | 0.80 | 0.77 | 0.70 | 0.54 | 0.81 | 0.79 | 0.79       | 0.87 | 0.80 | 0.87 | 0.80 | 0.70 | 0.70 |
| 11             | max           |           | 6.547 sec        |      |      |      |      |      |      |            |      |      |      |      |      |      |                 |      |      |      |      |      |      |            |      |      |      |      |      |      |
| 12             | min           |           | 5.300 sec        |      |      |      |      |      |      |            |      |      |      |      |      |      |                 |      |      |      |      |      |      |            |      |      |      |      |      |      |
| 13             | mean          |           | 5.073 sec        |      |      |      |      |      |      |            |      |      |      |      |      |      |                 |      |      |      |      |      |      |            |      |      |      |      |      |      |

Operating  
Down  
Turned off

In Figure 43, the average value of the OCR in the damage analysis was 0.80 meaning that the system's overall capability to recover from a single rupture was quite good, although the system still had room for improvement. The result has a few interesting patterns, one of which was that simulations 8, 18, 23, and 25 did not only yield the four lowest OCR values but also had all the same damage area, which was the mid-area of the system or more specifically, the bypass pipelines. This particular pattern in the analysis result clearly shows where to start for a more survivable and resilient system design. As a next step, a design engineer can create a number of design alternatives with different schemes of control strategy, valve placement, bypass pipeline placement, or service load locations, and then perform the same routine of modeling and damage analyses discussed above to the group of design alternatives in an automated manner.

### 3.2.2 Damage Simulation of CW-RSAD Surrogate Model

#### 3.2.2.1 CW-RSAD Model

In this section, the graph-based component surrogate modeling method will be applied to the CW-RSAD system for the damage simulation. The CW-RSAD model is a larger scale fluid system model, and Figure 44 shows the overall layout of the CW-RSAD fluid system, and the right side is the bow of the ship. The software model of the CW-RSAD is developed using Flowmaster V7. There is a slight difference between the layout shown in Figure 44 and the layout of the CW-RSAD Flowmaster model: in the original CW-RSAD model, both the AC plants are located in the aft of the ship.

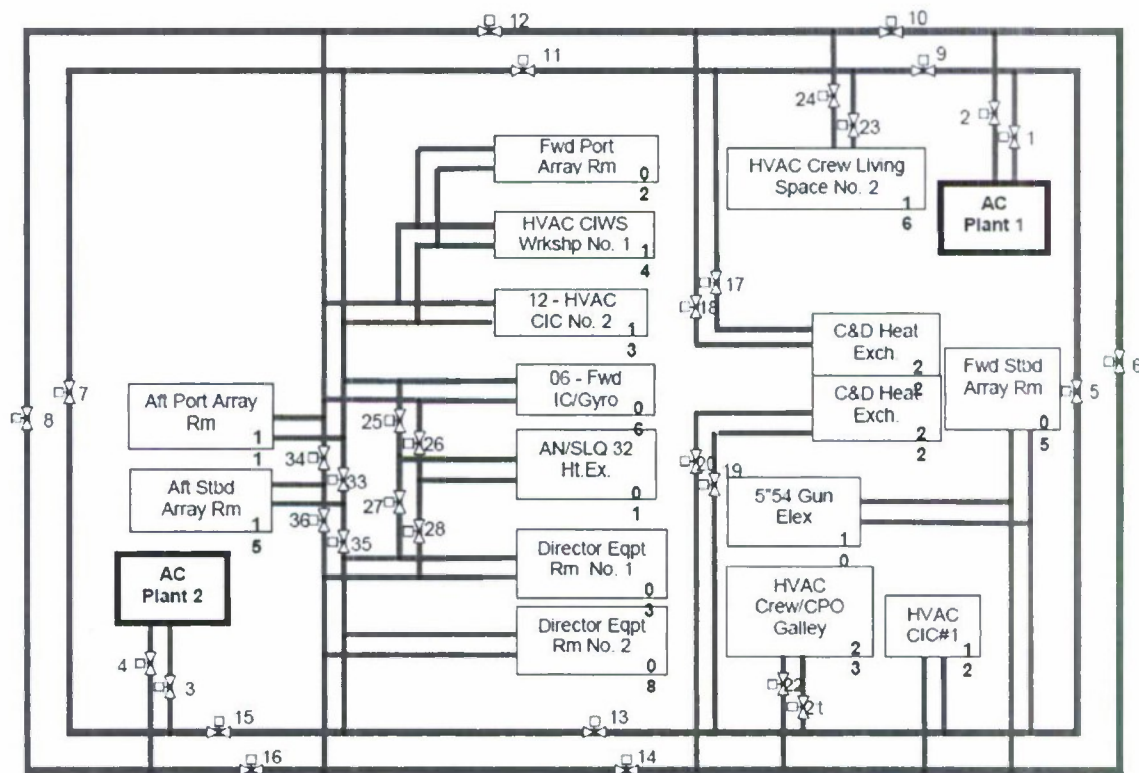


Figure 44: Thermal Service Loads Layout of CW-RSAD Notional Shipboard Fluid System



In order to achieve greater survivability and resilience of the shipboard engineering system, the CW-RSAD comprises redundant heat exchangers for the thermal service loads that are vital to ship operation such as CIC room HVAC, C&D system cooling, and director equipment room. As a result, the CW-RSAD has 16 heat exchangers that serve the 13 shipboard thermal service loads. In comparison, the notional YP fluid system has 7 heat exchangers and 6 service loads.

### 3.2.2.2 Graph Representation and Generation of CW-RSAD Surrogate Model

Similar to the application to the notional YP fluid system, the first step in this implementation is to define the graph representation of the CW-RSAD system, which is shown in Figure 45. This graph representation is based on the CW-RSAD Flowmaster model so the bow is on the left side of the graph model, and the two AC plants (PC components in the graph model) are located in the aft of the model.

To develop the CW-RSAD surrogate model, the component surrogate models created for the notional YP fluid model can be reused, which is one of the great advantages of the component surrogate modeling approach. The model types used for modeling the components are also shown in the graph representation in Figure 45. The specifications of the model types are listed in Table 3.

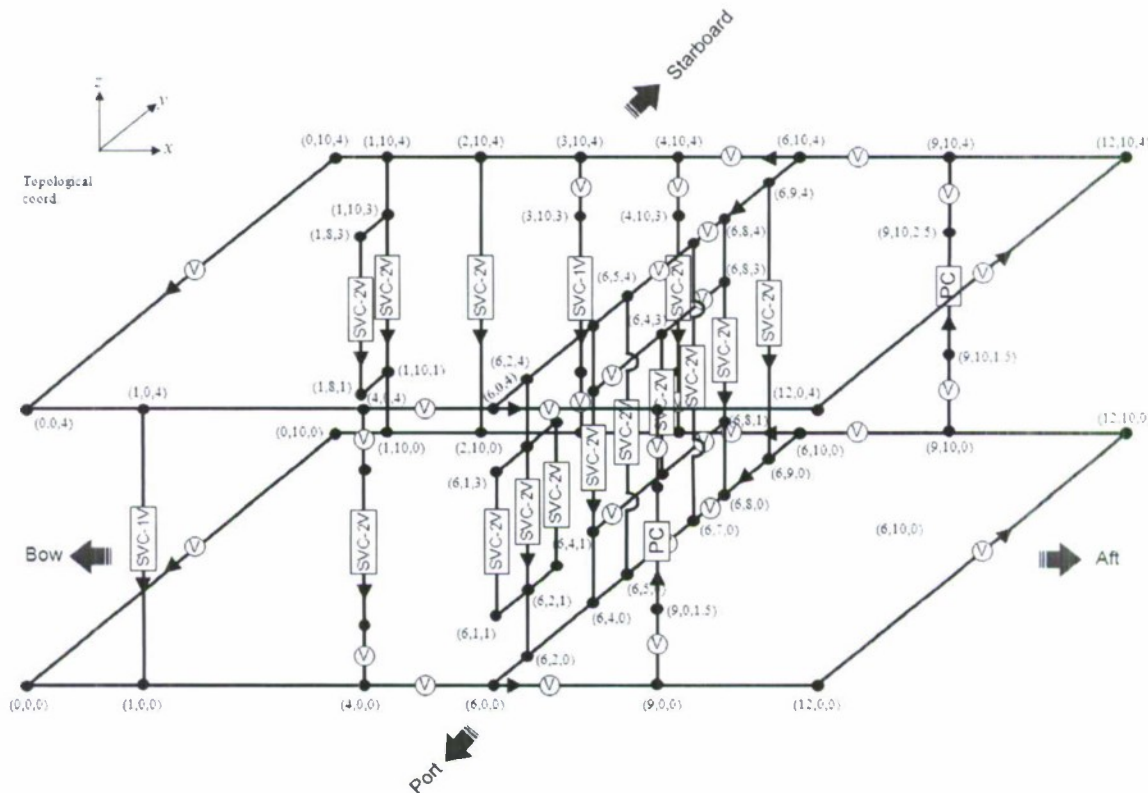


Figure 45: Graph Representation of CW-RSAD Flowmaster V7 Model

### 3.2.2.3 Damage Simulation

With the graph representation of the CW-RSAD model and the definition of the component models for the M&S algorithms implemented in Python, the incidence matrix and the reference damage control system model are auto-generated. In order to conduct the damage simulation, 48 damage bubble locations are defined in a way that considers all the damage possibilities uniformly. Simulation is run for each of the 48 damage cases as a damage scenario. The rest of the damage simulation setting is identical with that of the notional YP fluid model.

The system's damage recovery performance is also measured by the OCR, which is defined as follows:

$$OCR = \frac{A}{B} \quad (28)$$

Where,

$$A = w_1 \tilde{q}_1^f + w_2 \tilde{q}_2^f + w_3 \tilde{q}_3^f + w_4 \cdot \max(\tilde{q}_4^f, \tilde{q}_{10}^f) + w_5 \tilde{q}_5^f + w_6 \cdot \max(\tilde{q}_6^f, \tilde{q}_7^f) + w_7 \tilde{q}_8^f \\ + w_8 \tilde{q}_9^f + w_9 \tilde{q}_{11}^f + w_{10} \tilde{q}_{12}^f + w_{11} \cdot \max(\tilde{q}_{13}^f, \tilde{q}_{16}^f) + w_{12} \tilde{q}_{14}^f + w_{13} \tilde{q}_{15}^f$$

$$B = w_1 q_1^o + w_2 q_2^o + w_3 q_3^o + w_4 \cdot \max(q_4^o, q_{10}^o) + w_5 q_5^o + w_6 \cdot \max(q_6^o, q_7^o) + w_7 q_8^o \\ + w_8 q_9^o + w_9 q_{11}^o + w_{10} q_{12}^o + w_{11} \cdot \max(q_{13}^o, q_{16}^o) + w_{12} q_{14}^o + w_{13} q_{15}^o$$

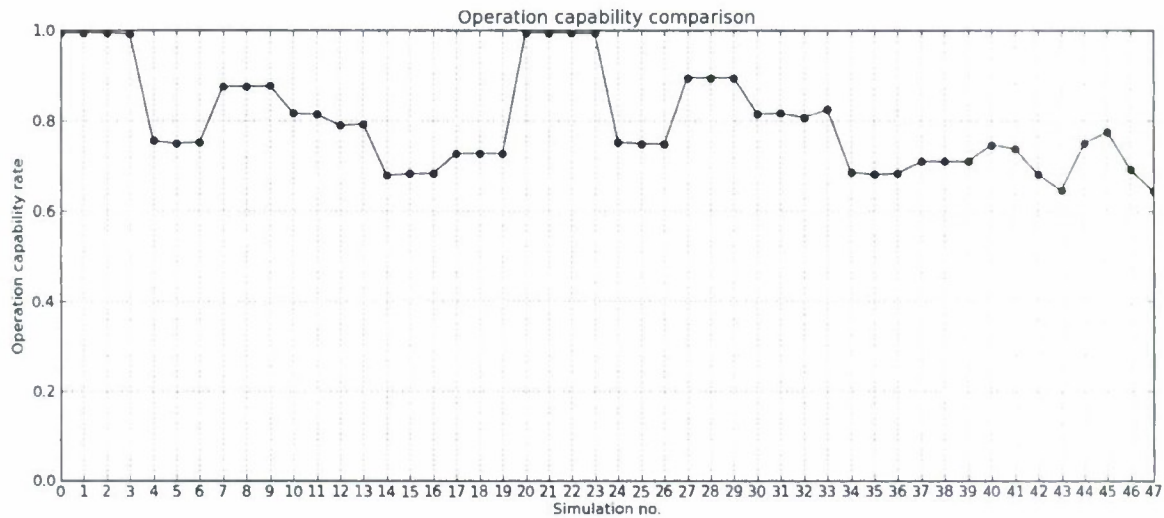
In Equation (28), all the notions are with the same as those used in Equations (26) and (27). The value of the weights  $w_j$  ( $j = 1, \dots, 13$ ) and the corresponding heat exchanger units, whose flow rates are  $q_i$  ( $i = 1, \dots, 16$ ), are defined in Table 6.

**Table 6: Service Load Weights and Mapping to Heat Exchanger Units**

| Service Load No. | Service Load     | Weight (1~10) | Heat Exchanger Assigned |
|------------------|------------------|---------------|-------------------------|
| 1                | HVAC Crew Rm     | 1             | HEX #1                  |
| 2                | Fwd Stbd Array   | 6             | HEX #2                  |
| 3                | 5.54 Elex Gun    | 5             | HEX #3                  |
| 4                | HVAC CIC         | 8             | HEX #4<br>HEX #10       |
| 5                | HVAC Crew/CPO    | 3             | HEX #5                  |
| 6                | C&D System       | 8             | HEX #6<br>HEX #7        |
| 7                | Fwd Port Array   | 6             | HEX #8                  |
| 8                | HVAC CIWS        | 8             | HEX #9                  |
| 9                | Fwd IC/Gyro      | 7             | HEX #11                 |
| 10               | AN/SLQ 32        | 6             | HEX #12                 |
| 11               | Director Eqpt Rm | 5             | HEX #13<br>HEX #16      |
| 12               | Aft Port Array   | 6             | HEX #14                 |
| 13               | Aft Stbd Array   | 6             | HEX #15                 |

Subsequently, the results of the damage simulations for the 48 damage cases are calculated and shown in Figure 46.





**Figure 46: Operational Capability Rates (OCR) of 48 Damage Cases**

In Figure 46, the simulations no. 0 to 19 cover the damages on the lower circular pipeline, the simulations no. 20 to 39 covers the damages on the upper circular pipeline, and those from no. 40 to 48 covers the damages on the mid-section bypass pipelines. Among the simulations no. 0 to 19, the ruptures of the first 10 simulations are located at the port side of the system, and the other 10 simulations are at the starboard side. As can be seen in Figure 46, the system generally has a lower recovery performance when there is a rupture at the starboard side, rather than the port side. Also, the results for the damages at the upper circular pipeline are very similar to those for the damages at the lower circular pipeline. The reason for this is that the configurations of the upper and lower pipelines are identical to each other.

Table 7 presents additional information of the simulation results, including the health status of the heat exchangers of the CW-RSAD system after the damage occurs. The green color indicates that a particular heat exchanger is in operating condition, the red color means it is in damaged condition, and the black color means that the unit is turned off.

Table 7 provides very useful information about failure recovery performance for different locations under consideration. For example, the simulations no.14 to 16 reveals that a rupture around the starboard mid-section area affects both of the redundant heat exchangers serving the same thermal service load "Director Equipment Room". This implies that the placement of this redundancy is ill-designed and need to be improved when the design is corrected. Similarly, the simulations no.4 to 6 and 10 to 14, and most of the simulations with mid-section damage yields relatively more thermal service failures than the rest. This also means that there is a high potential of system survivability benefit by improving design for those areas, such as placing a few more smart valves for effective damage isolation.

**Table 7: Component Failure Status, CW-RSAD Model**

**(a) Cases with Rupture Location in Upper Pipeline**

|                  |                   |                 | Rupture location |              |              |           |              |              |           |              |               |            |            |                 |               |            |               |               |            |               |                |             |
|------------------|-------------------|-----------------|------------------|--------------|--------------|-----------|--------------|--------------|-----------|--------------|---------------|------------|------------|-----------------|---------------|------------|---------------|---------------|------------|---------------|----------------|-------------|
|                  |                   |                 | Port lower       |              |              |           |              |              |           |              |               |            |            | Starboard lower |               |            |               |               |            |               |                |             |
| Simulation no    |                   |                 | 0                | 1            | 2            | 3         | 4            | 5            | 6         | 7            | 8             | 9          | 10         | 11              | 12            | 13         | 14            | 15            | 16         | 17            | 18             | 19          |
| Weight<br>(1~10) | Svc<br>Load<br>No | Compo-<br>nents | (0, 0, 0)        | (1 33, 0, 0) | (2 67, 0, 0) | (4, 0, 0) | (5 33, 0, 0) | (6 67, 0, 0) | (8, 0, 0) | (9 33, 0, 0) | (10 67, 0, 0) | (12, 0, 0) | (0, 10, 0) | (1 33, 10, 0)   | (2 67, 10, 0) | (4, 10, 0) | (5 33, 10, 0) | (6 67, 10, 0) | (8, 10, 0) | (9 33, 10, 0) | (10 67, 10, 0) | (12, 10, 0) |
|                  |                   | PC 1            |                  |              |              |           |              |              |           |              |               |            |            |                 |               |            |               |               |            |               |                |             |
|                  |                   | PC 2            |                  |              |              |           |              |              |           |              |               |            |            |                 |               |            |               |               |            |               |                |             |
| 1                | 1                 | HEX 1           |                  |              |              |           |              |              |           |              |               |            |            |                 |               |            |               |               |            |               |                |             |
| 6                | 2                 | HEX 2           |                  |              |              |           |              |              |           |              |               |            |            |                 |               |            |               |               |            |               |                |             |
| 5                | 3                 | HEX 3           |                  |              |              |           |              |              |           |              |               |            |            |                 |               |            |               |               |            |               |                |             |
| 8                | 4                 | HEX 4           |                  |              |              |           |              |              |           |              |               |            |            |                 |               |            |               |               |            |               |                |             |
|                  |                   | HEX 10          |                  |              |              |           |              |              |           |              |               |            |            |                 |               |            |               |               |            |               |                |             |
| 3                | 5                 | HEX 5           |                  |              |              |           |              |              |           |              |               |            |            |                 |               |            |               |               |            |               |                |             |
| 8                | 6                 | HEX 6           |                  |              |              |           |              |              |           |              |               |            |            |                 |               |            |               |               |            |               |                |             |
|                  |                   | HEX 7           |                  |              |              |           |              |              |           |              |               |            |            |                 |               |            |               |               |            |               |                |             |
| 6                | 7                 | HEX 8           |                  |              |              |           |              |              |           |              |               |            |            |                 |               |            |               |               |            |               |                |             |
| 8                | 8                 | HEX 9           |                  |              |              |           |              |              |           |              |               |            |            |                 |               |            |               |               |            |               |                |             |
| 7                | 9                 | HEX 11          |                  |              |              |           |              |              |           |              |               |            |            |                 |               |            |               |               |            |               |                |             |
| 6                | 10                | HEX 12          |                  |              |              |           |              |              |           |              |               |            |            |                 |               |            |               |               |            |               |                |             |
|                  |                   | HEX 13          |                  |              |              |           |              |              |           |              |               |            |            |                 |               |            |               |               |            |               |                |             |
| 5                | 11                | HEX 16          |                  |              |              |           |              |              |           |              |               |            |            |                 |               |            |               |               |            |               |                |             |
| 6                | 12                | HEX 14          |                  |              |              |           |              |              |           |              |               |            |            |                 |               |            |               |               |            |               |                |             |
| 6                | 13                | HEX15           |                  |              |              |           |              |              |           |              |               |            |            |                 |               |            |               |               |            |               |                |             |
|                  |                   | OCR             | 0.99             | 0.99         | 0.99         | 0.99      | 0.76         | 0.75         | 0.75      | 0.68         | 0.68          | 0.68       | 0.62       | 0.61            | 0.79          | 0.79       | 0.68          | 0.68          | 0.68       | 0.73          | 0.73           | 0.73        |


**(b) Cases with Rupture Location in Upper**

|               |               |             | Rupture location |           |               |               |           |               |               |           |               |                |                 |            |                |                |            |                |                |            |                |                 |             |  |  |  |  |
|---------------|---------------|-------------|------------------|-----------|---------------|---------------|-----------|---------------|---------------|-----------|---------------|----------------|-----------------|------------|----------------|----------------|------------|----------------|----------------|------------|----------------|-----------------|-------------|--|--|--|--|
|               |               |             | Port upper       |           |               |               |           |               |               |           |               |                | Starboard upper |            |                |                |            |                |                |            |                |                 |             |  |  |  |  |
| Simulation no | Weight (1-10) | Svc Load No | Compo-nents      | 20        | 21            | 22            | 23        | 24            | 25            | 26        | 27            | 28             | 29              | 30         | 31             | 32             | 33         | 34             | 35             | 36         | 37             | 38              | 39          |  |  |  |  |
|               |               |             |                  | (0, 0, 4) | (1, 33, 0, 4) | (2, 67, 0, 4) | (4, 0, 4) | (5, 33, 0, 4) | (6, 67, 0, 4) | (8, 0, 4) | (9, 33, 0, 4) | (10, 67, 0, 4) | (12, 0, 4)      | (0, 10, 4) | (1, 33, 10, 4) | (2, 67, 10, 4) | (4, 10, 4) | (5, 33, 10, 4) | (6, 67, 10, 4) | (8, 10, 4) | (9, 33, 10, 4) | (10, 67, 10, 4) | (12, 10, 4) |  |  |  |  |
|               |               |             | PC 1             |           |               |               |           |               |               |           |               |                |                 |            |                |                |            |                |                |            |                |                 |             |  |  |  |  |
|               |               |             | PC 2             |           |               |               |           |               |               |           |               |                |                 |            |                |                |            |                |                |            |                |                 |             |  |  |  |  |
| 1             | 1             | 1           | HEX 1            |           |               |               |           |               |               |           |               |                |                 |            |                |                |            |                |                |            |                |                 |             |  |  |  |  |
| 6             | 2             | 2           | HEX 2            |           |               |               |           |               |               |           |               |                |                 |            |                |                |            |                |                |            |                |                 |             |  |  |  |  |
| 5             | 3             | 3           | HEX 3            |           |               |               |           |               |               |           |               |                |                 |            |                |                |            |                |                |            |                |                 |             |  |  |  |  |
| 8             | 4             | 4           | HEX 4            |           |               |               |           |               |               |           |               |                |                 |            |                |                |            |                |                |            |                |                 |             |  |  |  |  |
| 3             | 5             | 5           | HEX 10           |           |               |               |           |               |               |           |               |                |                 |            |                |                |            |                |                |            |                |                 |             |  |  |  |  |
| 8             | 6             | 6           | HEX 6            |           |               |               |           |               |               |           |               |                |                 |            |                |                |            |                |                |            |                |                 |             |  |  |  |  |
| 6             | 7             | 7           | HEX 7            |           |               |               |           |               |               |           |               |                |                 |            |                |                |            |                |                |            |                |                 |             |  |  |  |  |
| 8             | 8             | 8           | HEX 8            |           |               |               |           |               |               |           |               |                |                 |            |                |                |            |                |                |            |                |                 |             |  |  |  |  |
| 7             | 9             | 9           | HEX 9            |           |               |               |           |               |               |           |               |                |                 |            |                |                |            |                |                |            |                |                 |             |  |  |  |  |
| 6             | 10            | 10          | HEX 11           |           |               |               |           |               |               |           |               |                |                 |            |                |                |            |                |                |            |                |                 |             |  |  |  |  |
| 5             | 11            | 11          | HEX 12           |           |               |               |           |               |               |           |               |                |                 |            |                |                |            |                |                |            |                |                 |             |  |  |  |  |
| 6             | 12            | 12          | HEX 13           |           |               |               |           |               |               |           |               |                |                 |            |                |                |            |                |                |            |                |                 |             |  |  |  |  |
| 6             | 13            | 13          | HEX 14           |           |               |               |           |               |               |           |               |                |                 |            |                |                |            |                |                |            |                |                 |             |  |  |  |  |
|               |               |             | HEX 15           |           |               |               |           |               |               |           |               |                |                 |            |                |                |            |                |                |            |                |                 |             |  |  |  |  |
|               |               |             | OCR              | 0.99      | 0.99          | 0.99          | 0.99      | 0.75          | 0.75          | 0.75      | 0.69          | 0.69           | 0.69            | 0.62       | 0.62           | 0.61           | 0.62       | 0.69           | 0.68           | 0.68       | 0.71           | 0.71            | 0.71        |  |  |  |  |

Pipeline

**(c) Cases with Rupture Location in Mid-Section Bypass Pipeline**

|                |             |             | Rupture location |           |           |           |           |           |           |           |
|----------------|-------------|-------------|------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
|                |             |             | Mid lower        |           |           |           | Mid upper |           |           |           |
| Simulation no. |             |             | 40               | 41        | 42        | 43        | 44        | 45        | 46        | 47        |
| Weight (1~10)  | Svc Load No | Compo-nents | (6, 2, 0)        | (6, 4, 0) | (6, 6, 0) | (6, 8, 0) | (6, 2, 0) | (6, 4, 0) | (6, 6, 0) | (6, 8, 0) |
|                |             | PC 1        |                  |           |           |           |           |           |           |           |
|                |             | PC 2        |                  |           |           |           |           |           |           |           |
| 1              | 1           | HEX 1       |                  |           |           |           |           |           |           |           |
| 6              | 2           | HEX 2       |                  |           |           |           |           |           |           |           |
| 5              | 3           | HEX 3       |                  |           |           |           |           |           |           |           |
| 8              | 4           | HEX 4       |                  |           |           |           |           |           |           |           |
|                |             | HEX 10      |                  |           |           |           |           |           |           |           |
| 3              | 5           | HEX 5       |                  |           |           |           |           |           |           |           |
| 8              | 6           | HEX 6       |                  |           |           |           |           |           |           |           |
|                |             | HEX 7       |                  |           |           |           |           |           |           |           |
| 6              | 7           | HEX 8       |                  |           |           |           |           |           |           |           |
| 8              | 8           | HEX 9       |                  |           |           |           |           |           |           |           |
| 7              | 9           | HEX 11      |                  |           |           |           |           |           |           |           |
| 8              | 10          | HEX 12      |                  |           |           |           |           |           |           |           |
| 5              | 11          | HEX 13      |                  |           |           |           |           |           |           |           |
|                |             | HEX 16      |                  |           |           |           |           |           |           |           |
| 6              | 12          | HEX 14      |                  |           |           |           |           |           |           |           |
| 6              | 13          | HEX 15      |                  |           |           |           |           |           |           |           |
| OCR            |             |             | 0.75             | 0.74      | 0.68      | 0.66      | 0.75      | 0.77      | 0.69      | 0.64      |


 Operating  
 Down  
 Turned-off

### 3.3 Future Works

The future works involve further refining the graph-based component surrogate modeling method and expanding its application domain to the DC power systems. In order to achieve this goal, both the current method and M&S environment, which are developed heavily based on the application of fluid system, need to be revamped and reorganized. Additional development of simulation schemes that are more suitable to the simulation of DC power systems may be necessary. DC power component models will be generated by either physics-based modeling or surrogate modeling, depending on the computational demand of the power model.

More extensively, thermal modeling modules must be developed in order to investigate the interactions between the DC power model and the fluid model. As for the test and evaluation of the extended capability of the graph-based modeling method, a simulation-based design analysis will be performed in a similar way to the demonstration of the damage analysis for the fluid system presented in the previous research.

Along with the modeling for the DC power system, a new or modified damage generation method will be formulated for the damage analysis. In the current damage generation approach, the probability of damage at different locations of the notional ship was not taken into account for the simplicity purpose. In this newly formulated damage analysis, the location and the size of each damage entity will be determined through a probabilistic analysis, which will be based on the historical data of naval surface combatants and engineers' intuition.

### Publications

1. Moon, K and Mavris, D, "Modeling and Simulation for Damage Analysis of Intelligent Self-Reconfigurable Ship Fluid Systems in Early Design Phase," under review, submitted to *Simulation Modelling Practice and Theory, Elsevier* in Jun. 10, 2010



2. Michael Balchanos, Yongchang Li, Dimitri Mavris, "A Theoretical Framework for the Analysis and Design of Resilient Engineering Systems", To be submitted to the Journal for Safety Research, by Elsevier, 2010 (In progress)
3. Matt Hoepfer, Yongchang Li, Dimitri Mavris, "Integration of a Dynamic Modeling and simulation Environment for Naval Complex System Design", To be submitted to the *Simulation Modeling Practice and Theory*, Elsevier, 2011 (In progress)

## References

- [1] Ascher, U. M., Petzold, L. R., "Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations", SIAM: Society for Industrial and Applied Mathematics, 1998, ISBN-10: 0898714125, ISBN-13: 978-0898714128
- [2] Ball, R. E., "The Fundamentals of Aircraft Combat Survivability Analysis and Design", AIAA (American Institute of Aeronautics & Astronautics), 2003.
- [3] Ball, R. E. & Atkinson, D. B., "A History of the Survivability Design of Military Aircraft", *Naval Postgraduate School Report, AIAA*, 1998, 16.
- [4] Boyd, James E., "Strength of Materials", McGraw-Hill, 1917.
- [5] Burden, R. L., Faires, J. D. (2002), "Numerical Analysis", 3rd Edition, Brooks Cole 2002, ISBN-10: 0534407617, ISBN-13: 978-0534407612
- [6] Crouzet, N. and Turinsky, P. J., "A second-derivative-based adaptive time-step method for spatial kinetics calculations", *Nuclear science and engineering* 1996, vol123, iss. 2, pp. 206 - 214
- [7] Dijkstra, A., Resilience Engineering and Safety Management Systems in Aviation, 2007.
- [8] Doerry, N.H., H. Fireman, "Designing All Electric Ships," Presented at IMDC 2006, Ann Arbor, Michigan, 16-19 May 2006.
- [9] Fehlberg, E., 1969, "Low-order classical Runge-Kutta formulas with step size control and their application to some heat transfer problems". NASA Technical Report 315.
- [10] Fehlberg, E. 1970, "Klassische Runge-Kutta-Formeln vierter und niedrigerer Ordnung mit Schrittwelten-Kontrolle und ihre Anwendung auf Wärmeleitungsprobleme," *Computing (Arch. Elektron. Rechnen)*, vol. 6, pp. 61-71.
- [11] Hanifari, D. T., Navy system effectiveness manual NAVMAT p3941-b proposed, "Technical document NAVMAT P3941-B, Jul 1971-Sep 1973.
- [12] Harriss, R., Hohenemser, C., Kates, R. , Goodman, G. & Rowe, W. (ed.), "Energy risk management", *Academic Press*, 1979, 103-138.
- [13] Holling, C., "Resilience and stability of ecological systems", *Annual review of ecology and systematics*, Annual Reviews, 1973, 4, 1-23.
- [14] Hollnagel, E., Woods, D. & Leveson, N., "Resilience engineering: Concepts and precepts", Ashgate Pub Co, 2006.
- [15] Joukhadar, A., and Laugier, C., "Adaptive time step for fast converging dynamic simulation system", *Proceedings of the 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Volume 2, 4-8 Nov 1996 Page(s):418 – 424
- [16] Laracy, J., Leveson, N., "Applying STAMP to Critical Infrastructure Protection", 2007.
- [17] Leveson N. L., "Role of Software in Spacecraft Accidents", *Journal Of Spacecraft And Rockets*, 2004, 41, 564-575.

- [18]Leveson, N., "White Paper on Approaches to Safety Engineering," 2003.
- [19]Leveson, N., "Safeware: system safety and computers", *ACM New York, NY, USA*, 1995.
- [20]Lively, K.A., D.H. Scheidt, and K.F. Drew, "Mission Based Engineering Plant Control," *Reconfiguration and Survivability Symposium*, 2005
- [21]McManus H., Richards M., et. al, "A Framework for Incorporating "ilities" in Tradespace Studies", *AIAA Journal*, 2005.
- [22]Mehrkanoon, S., Suleiman, M., Zanariah, A., M., Khairil, I. O. (2009), "Parallel Solution in Space of Large ODEs Using Block Multistep Method with Step Size Controller", *European Journal of Scientific Research* ISSN 1450-216X Vol.36 No.3 (2009), pp 491-501
- [23]Mitchell, S., Mannan, M., and O'Connor, M., "Designing resilient engineered systems," *Chemical Engineering Progress*, vol. 102, no. 4, pp. 39{45, 2006.
- [24]Moon, K, "Self-Reconfigurable Ship Fluid-Network Modeling for Simulation-Based Design," Ph.D. thesis, Georgia Institute of Technology, Atlanta, Georgia, Aug. 2010
- [25]Nairouz, B., Hoepfer, M., Weston, N., and Mavris, D., "Investigations for Time Step Settings in a Dynamic System Co-Simulation Environment", *Electric Ship Design Symposium*, Washington D.C., Jan 2009
- [26]Richards, M., "Multi-Attribute Tradespace Exploration for Survivability", Engineering Systems Division, Massachusetts Institute of Technology, 2009.
- [27]Savcenco, V., Mattheij, R. M. M. (2010), "Multirate Numerical Integration for Stiff ODEs", *Mathematics in Industry*, 2010, Volume 15, Part 2, 327-332, DOI: 10.1007/978-3-642-12110-4\_50
- [28]Vanderplaats, G. N.: "Numerical Optimization Techniques for Engineering Design", 4th Edition, Vanderplaats Research & Development, Inc., 2005, ISBN 0-944956-03-3
- [29]Vugrin, E., Warren, D., Ehlen, M., and Camphouse, R., "A framework for assessing the resilience of infrastructure and economic systems," *Sustainable and Resilient Critical Infrastructure Systems: Simulation, Modeling, and Intelligent Engineering*, p. 77, 2010.
- [30]Xiang, Y., Poole, D., Beddoes, M. P., 1993. Multiply sectioned Bayesian networks and junction forests for large knowledge-based systems. *Computational Intelligence* 19.
- [31]Yarbrough, N.R., Russell E.K., "The Joint Command and Control Ship (JCC(X)) Approach to Survivability Requirements Development: Total Ship Survivability Assessment," Association of Scientists and Engineers – 38th Annual Technical Symposium, May 9, 2002.
- [32]U.S. Navy, Yard Patrol Craft YP data sheet, available online at: <http://www.dt.navy.mil/tot-shi-sys/com-cra/pro-gal/yp> , accessed October 2009.
- [33]U.S Navy, Data Sheets on YP-679, available online at: <http://www.navy.mil/navydata/>], accessed October 2009.
- [34]U.S. Naval Academy, "EE 301, Shipboard Power Distribution Systems", lecture notes, retrieved October 2009.
- [35]Wendland, H., "Scattered Data Approximation", Cambridge University Press, 2005, ISBN-10: 0521131014, ISBN-13: 978-0521131018
- [36]Zeigler, B., Kim, T. G., and Praehofer, H, "Theory of Modeling and Simulation – Second Edition", New York, Academic Press, 2000